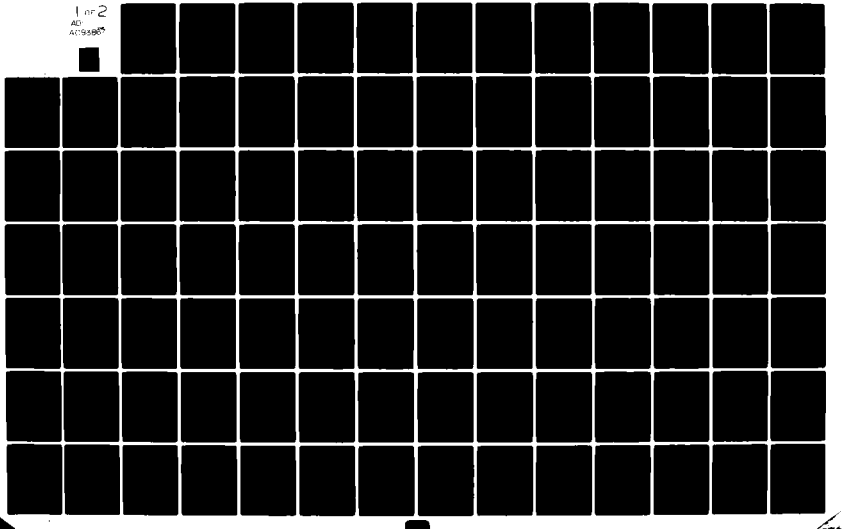
AD-A093 867 TRACOR APPLIED SCIENCES AUSTIN TX F/G 17/1
HYBRID PASSIVE TRACKING ALGORITHMS.(U)
OCT 80 G CORSER, T WILSON N00014-78-C-0670
UNCLASSIFIED NL



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (6)	2. GOVT ACCESSION NO. AD-A093867	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) FINAL REPORT: HYBRID PASSIVE TRACKING ALGORITHMS (19)	5. TYPE OF REPORT & PERIOD COVERED (9) FINAL Rept'y
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) (10) Glenn Corser & Thomas Wilson	8. CONTRACT OR GRANT NUMBER(s) (15) N00014-78-C-0670
9. PERFORMING ORGANIZATION NAME AND ADDRESS Tracor, Inc., 6500 Tracor Lane, Austin, TX 78721	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 65152N, RØ145 TW, NR 274-299
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Analysis Program (Code 431) Office of Naval Research, Arlington, VA 22217	12. REPORT DATE October 31, 1980
	13. NUMBER OF PAGES 117
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (11) 31 (12) 92	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED (17) RØ145 TW

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report is concerned with the refinement and performance evaluation of a passive hybrid tracking algorithm. It is composed of a simplified batch filter and a numerically-stable extended Kalman filter, linked together by a set of statistically meaningful switching rules. Comparisons in performance between the hybrid, Tracor's MLP algorithm, and a sequential-sequential algorithm (where the batch initializer is replaced with an iterated extended Kalman Filter) are made using simulated data.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

41 124

(12)

ONR Contract Number N00014-78-C-0670
Task Number 247-299
Tracor Project 038-016

FINAL REPORT

HYBRID PASSIVE TRACKING ALGORITHMS (U)

Submitted to:

Scientific Officer
Operations Research Analyst
Mathematical and Information Sciences Division
800 North Quincy Street
Arlington, Virginia 22207

Attention: James G. Smith
Code 431

31 October 1980

Submitted by:

Glenn Corser
Engineer/Scientist

Approved by:

Thomas Wilson
Project Director

Tracor Applied Sciences

Tracor, Inc. 6500 Tracor Lane Austin, Texas 78721 Telephone 512: 926 2800

TABLE OF CONTENTS

Section Number		Page Number
	LIST OF FIGURES	iii
1.0	INTRODUCTION	1
1.1	Background - Processing Alternatives	1
1.2	Hybrid Research Effort and Program Objectives	3
1.3	Summary of Results	4
2.0	THEORY AND EXPLANATION OF ALGORITHMS	5
2.1	Maximum Likelihood Procedure	5
2.1.1	Motion Models	5
2.1.2	MLP Algorithm	9
2.2	Hybrid Algorithm	19
2.2.1	Introduction	19
2.2.2	Aspects of Sequential Filter	20
2.2.3	Aspects of the Batch Filter	34
2.2.4	Switching Rules	38
2.3	Iterated Sequential Algorithm	46
2.3.1	Aspects of Iterated Sequential Algorithm	46
3.0	TEST RESULTS (COMPARISONS)	50
3.1	Introduction	50
3.2	Scenario Descriptions	52
3.3	Generation of Data for DIFAR Simulation	53

TABLE OF CONTENTS (Continued)

Section Number		Page Number
3.3.1	General Approach	53
3.3.2	Analysis	60
3.3.3	Summary	67
3.3.4	Signal-to-Noise Ratios and Variances	68
3.4	Test Results and Analysis	73
3.4.1	Introduction	73
3.4.2	Scenario One	98
3.4.3	Scenario Two	101
3.4.4	Scenario Three	102
3.4.5	Scenario Four	103
4.0	CONCLUSIONS AND RECOMMENDATIONS	104
4.1	Conclusions	104
4.2	Recommendations	105
5.0	MULTIPLE TARGET PROBLEM	109
6.0	REFERENCES	116

LIST OF FIGURES

Figure Number		Page Number
2.1	PROGRAM FLOW FOR MANEUVERING TARGET TRACKING	8
2.2a	GIVENS' ROTATIONS	15
2.2b	SQUARE ROOT FORMULATION OF GIVENS' ROTATIONS	16
2.2.1	HYBRID ALGORITHM MACRO STRUCTURE	21
2.2.2	LOGIC DIAGRAM - KALMAN FILTER	28
2.2.3	KALMAN FILTER EQUATIONS	30
2.2.4	FLOWCHART - HYBRID BATCH INITIALIZER	36
2.2.4.1	FLOWCHART - BATCH SWITCHING LOGIC	42
2.3.1	FLOWCHART - ITERATED SEQUENTIAL INITIALIZER	48
3.2a	SCENARIO 1 DESCRIPTION	54
3.2b	SCENARIO 2 DESCRIPTION	55
3.2c	SCENARIO 3 DESCRIPTION	56
3.2d	SCENARIO 4 DESCRIPTION	57
3.2e	COMMON SCENARIO INPUT AND ENVIRONMENTAL VARIABLES	58
3.3.2.1	GENERAL PROCESSOR FOR DIFAR BUOY	59
3.3.3	FLOWCHART OF PROCESSOR	69
3.4.1a	HYBRID - SCENARIO 1	74
3.4.1b	MLP - SCENARIO 1	75
3.4.1c	SEQUENTIAL - SCENARIO 1	76
3.4.2a	HYBRID - SCENARIO 2	77

LIST OF FIGURES (Continued)

Figure Number		Page Number
3.4.2b	MLP - SCENARIO 2	78
3.4.2c	SEQUENTIAL - SCENARIO 2	79
3.4.3a	HYBRID - SCENARIO 3	80
3.4.3b	MLP - SCENARIO 3	81
3.4.3c	SEQUENTIAL - SCENARIO 3	82
3.4.4a	HYBRID - SCENARIO 4	83
3.4.4b	MLP - SCENARIO 4	84
3.4.4c	SEQUENTIAL - SCENARIO 4	85
3.4.5a	HYBRID - SCENARIO 1	86
3.4.5b	MLP - SCENARIO 1	87
3.4.5c	SEQUENTIAL - SCENARIO 1	88
3.4.6a	HYBRID - SCENARIO 2	89
3.4.6b	MLP - SCENARIO 2	90
3.4.6c	SEQUENTIAL - SCENARIO 2	91
3.4.7a	HYBRID - SCENARIO 3	92
3.4.7b	MLP - SCENARIO 3	93
3.4.7c	SEQUENTIAL - SCENARIO 3	94
3.4.8a	HYBRID - SCENARIO 4	95
3.4.8b	MLP - SCENARIO 4	96
3.4.8c	SEQUENTIAL - SCENARIO 4	97
3.4.9	AVERAGE DISTANCE ERROR VERSUS SCENARIO	99
3.4.10	AVERAGE EXECUTION TIME VERSUS SCENARIO	100
5.1	MULTI-TARGET ALGORITHM USING CLUSTER ANALYSIS	112

1.0 INTRODUCTION

A modern nuclear submarine radiates acoustic energy as it moves underwater in the ocean. It is possible to detect and track these submarines by using this radiated energy. Typically, this energy is acquired by passive sonobuoys and processed electronically to give estimates of the Doppler-shifted frequency associated with the submarine and also an estimated bearing.

1.1 Background - Processing Alternatives

Once the raw data from the sonobuoys has been processed to give frequency and bearing estimates, it is passed through a tracking algorithm which will produce estimates of the target's state vector, i.e., position, velocity, etc.

At present there are two primary classes of algorithms for solving this problem--batch algorithms and sequential algorithms. The names are derived from the way each algorithm processes data. Batch algorithms simultaneously process several data points (i.e., a batch of data) at once to provide tracking parameters, while sequential algorithms process one data point at a time, giving updated state vector estimates after each data point has been processed. Over the past several years, Tracor has developed several batch algorithms called Maximum Likelihood Procedures (MLP) to track submarine targets for a number of specialized tracking system configurations. Each of these procedures has been tested on sets of real-world and simulated data with varying degrees of success. However, in doing so several points became clear:

(1) The MLP is self-initializing. That is, good initial estimates of the state vector can be generated from the data. There is no need to supply an accurate initial guess.

(2) The MLP requires several different motion models to characterize possible submarine maneuvers. Model selection can become a problem in terms of time and appropriateness.

(3) Because the MLP processes data in a batch, extremely accurate estimates of the state vector can be obtained when the proper motion model has been selected. However, serious loss of track can occur when the model currently being used no longer becomes valid but the selection process has not yet selected a new model.

(4) The algorithm possesses moderate to substantial computer time requirements, depending upon the complexity of the track being attempted, and moderate storage requirements.

To provide a means of comparison for the MLP and also in an attempt to gain insight into the nature of sequential algorithms, Tracor, in conjunction with Dr. Byron Tapley of the University of Texas, designed and implemented a sequential tracking algorithm based on the extended Kalman filter. While testing this algorithm several things became clear:

(1) This algorithm has smaller core storage and execution time requirements than the MLP.

(2) Tracking accuracy is as good or better than the MLP.

(3) The sequential algorithm is highly susceptible to initialization errors, requiring a fairly close initial state vector estimate to maintain good track.

(4) The sequential model can be implemented with a single stochastic motion model. This eliminates the need for the model selection process, and also allows some "slack" in estimating the tracking parameters.

1.2 Hybrid Research Effort and Program Objectives

After testing of the MLP and sequential algorithms revealed their complementary strengths, it was then decided that a combination of the two approaches, or a hybrid algorithm, should be developed to take advantage of these strengths. It was hoped that the MLP algorithm could intialize the tracking procedure for the sequential algorithm and also provide reinitialization should the sequential filter lose track. A preliminary version of the hybrid was created using the MLP and sequential algorithms and early tests showed potential. However, several problems remained. Among them were:

(1) The MLP was too cumbersome to use efficiently as an initializer; a more efficient batch procedure was needed.

(2) Information transfer between the batch and sequential portions had to be streamlined and made more efficient.

(3) The sequential algorithm's numerical properties needed to be improved for smaller word length computers.

(4) The capability for handling maneuvering sensors was needed.

(5) The capability for tracking multiple targets was needed.

(6) Improvement and refinement of the switching rules from the batch-to-sequential and sequential-to-batch modes of operation were needed.

1.3 Summary of Results

A hybrid algorithm incorporating the above features has been designed and implemented. This study provides a comparison between the hybrid algorithm and the MLP and, in addition, a comparison with a sequential algorithm that contains an iterated sequential starter.

2.0 THEORY AND EXPLANATION OF ALGORITHMS

2.1 Maximum Likelihood Procedure (MLP)

2.1.1 Motion Models

The MLP algorithm is based on the assumption that submarines typically execute a fairly small set of maneuvers, including:

(1) Constant Linear Velocity

(2) Constant Linear Acceleration

(3) Constant Radius and Angular Velocity Turns

(4) Constant Radius and Angular Acceleration Turns

Each of these maneuvers has a corresponding motion model. They are:

(1) Constant Velocity

$$X(t) = X_o + V_x t$$

$$Y(t) = Y_o + V_y t$$

$$V_x(t) = V_x$$

$$V_y(t) = V_y$$

(2) Constant Acceleration

$$X(t) = X_o + V_x t + .5\ a_x t^2$$

$$Y(t) = Y_o + V_y t + .5\ a_y t^2$$

$$V_x(t) = V_x + a_x t$$

$$V_y(t) = V_y + a_y t$$

(3) Constant Speed Turn

$$X(t) = X_o + V_x \sin(wt)/w - V_y[1 - \cos(wt)]/w$$

$$Y(t) = Y_o + V_x[1 - \cos(wt)]/w + V_y \sin(wt)/w$$

$$V_x(t) = V_x \cos(wt) - V_y \sin(wt)$$

$$V_y(t) = V_x \sin(wt) + V_y \cos(wt)$$

(4) Constant Angular Acceleration, Constant Radius Turn

$$X(t) = X_o + \{V_x \sin wt(1+.5at) - V_y[1-\cos wt(1+.5at)]\}/w$$

$$Y(t) = Y_o + \{V_x[1-\cos wt(1+.5at) + V_y \sin wt(1+.5at)]\}/w$$

$$V_x(t) = (1+at)\{V_x \cos wt(1+.5at) - V_y \sin wt(1+.5at)\}$$

$$V_y(t) = (1+at)\{V_x \sin wt(1+.5at) + V_y \cos wt(1+.5at)\}$$

Each of these maneuvers is characterized by a unique set of target parameters. For example, constant velocity motion is characterized by the following vector:

$$\gamma = [X_t, Y_t, V_{x_t}, V_{y_t}, f_1, \ldots, f_n]$$

where

X_t = X-position at time, t

Y_t = Y-position at time, t

V_{x_t} = X-velocity at time, t

V_{y_t} = Y-velocity at time, t

$f_1, \ldots, f_n$ = Source frequency parameter for each target source frequency

The approach used by MLP to track targets is summarized in Figure 2-1. Frequency and bearing data from each sonobuoy maintaining contact with the target are input to each maneuver model for which there is a non-zero a priori probability that the target could be executing that maneuver. The Maximum Likelihood procedure is applied to estimating the parameters associated with each model; these parameter values and the residual data errors are compared for each of the models and three decisions are reached.

(1) The maneuver model and parameter vector associated with that model are selected.

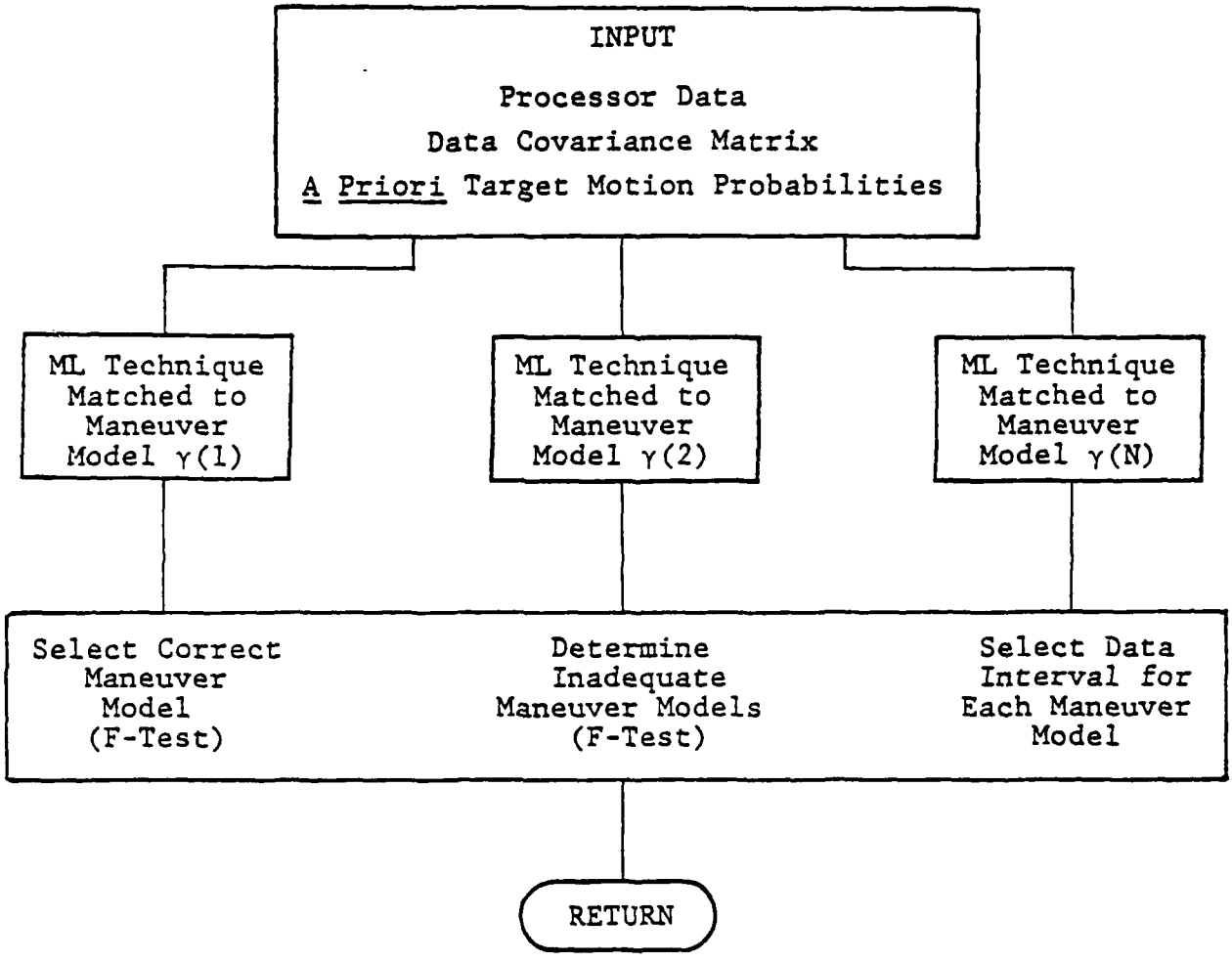


FIG. 2.1 - PROGRAM FLOW FOR MANEUVERING TARGET TRACKING

(2) The maneuver models which are totally inadequate to explain the data are determined.

(3) The reduced tracking intervals for each of the models which could explain the data are selected.

At the conclusion of this procedure new data are input and the procedure is repeated. Note that the MLP passes the data through four Maximum Likelihood parameter estimation algorithms, each one corresponding to one of the maneuver models described above.

2.1.2 MLP Algorithm

The mathematical structure of the Maximum Likelihood procedure is the same regardless of the motion model and data types used. Associated with each measurement is a nonlinear function that generates an estimate of that data measurement given the time of the measurement, a set of motion parameters, and a motion model. For example, if a frequency measurement is taken using sensor, j, at time, t_i, the estimated frequency is given by,

$$f(t_i,\vec{X}) = f_0/[1+\vec{V}(t_i)\cdot\vec{R}_j(t_i)/c|\vec{R}_j(t_i)|],$$

where

$\vec{X}$ = motion parameter vector,

$\vec{V}(t_i)$ = velocity vector of target at time, t_i,

$\vec{R}_j(t_i)$ = range vector of the target from sensor, j, at time, t_i

f_0 = center frequency

c = velocity of propagation of sound

If a bearing measurement is considered, then

$$f(t_i,\vec{Y}) = \tan^{-1}[Y_j(t_i)/X_j(t_i)]$$

where

$X_j(t_i)$ = X-coordinate of target at time, t_i, relative to sensor, j

$Y_j(t_i)$ = Y-coordinate of target at time, t_i, relative to sensor, j

The program can also use range, Doppler ratio, Doppler difference, and time difference of arrival. The appropriate motion model is used to generate $\vec{R}_j(t)$ and $\vec{V}_j(t)$ for any time, t.

Let $E_i = Z_i - f(t_i,\vec{Y})$. It is desired that the f's be close to the measurement values Z_i. A reasonable measure to consider for this closeness is the sum of squares of residuals for a given parameter vector,

$$S(\vec{Y}) = \sum_{i=1}^{N} E_i^2/\sigma_i^2,$$

where σ_i^2 is the measurement error of the observation, Z_i. If

the above sum is small, the f's will be close to the Z_i's. If it is assumed that the measurement values are perturbed from the true mean values of $f(t_i,\vec{\gamma})$ by Gaussian random variables, the minimization of the sum is equivalent to maximizing the likelihood function $L(\Omega)$ of the sample Z as

$$L(\Omega) = \text{EXP}\left[-\sum_{i=1}^{N} E_i^2/\sigma_i^2\right] / \left[(2\pi)^{N/2} \prod_{i=1}^{N} \sigma_i\right] .$$

Even if the measurement errors are not Gaussianly distributed, this procedure provides a least squares fit to the data. Thus, the problem statement is: Given a set of measurements, Z, and a motion model, find the motion parameter vector, $\vec{\gamma}$, that minimizes

$$\sum_{i=1}^{N} E_i^2/\sigma_i^2 .$$

Since $f(t_i,\vec{\gamma})$ is a nonlinear function of the vector, $\vec{\gamma}$, a direct solution is not, in general, possible. However, a procedure may be used in which a linear approximation is applied iteratively to search for a solution. If the nonlinear problem is sufficiently well behaved the iterative technique will converge to the desired solution. In general, it is not possible or is too difficult to prove whether this technique will converge in a given problem. Hence, the algorithm is executed. If it converges, a solution is found; if not, add more data and try again.

To linearize the problem each function, f, is expanded in a first order Taylor series about an assumed parameter vector, $\vec{\gamma}_o$, as,

$$f(t_i,\vec{\gamma}) = f(t_i,\vec{\gamma}_o) + \sum_{k=1}^{M} \delta_k \frac{\partial f(t_i,\vec{\gamma})}{\partial \gamma_k}\bigg|_{t_i,\vec{\gamma}_o}$$

where δ_k is the variation of the $k^{\underline{th}}$ parameter of $\vec{\gamma}$. If the vector, E_o, is defined by the elements,

$$\vec{E}_o = Z_i - f(t_i,\vec{\gamma}_o)$$

and the matrix, X, by the elements

$$X_{ij} = \frac{\partial f(t,\vec{\gamma})}{\partial \vec{\gamma}_j}\bigg|_{t_i,\vec{\gamma}_o}$$

Then, the linear approximation problem is

$$X\delta = \varepsilon_o \ .$$

The vector, $\vec{\delta}$, may be found in the usual way for minimizing the weighted sum of squares of residuals (Reference 1). The normal equations are

$$X^T\sigma^{-1}X\delta = X^T\sigma^{-1}\varepsilon_o$$

and δ is found to be

$$\delta = (X^T\sigma^{-1}X)^{-1}X^T\sigma^{-1}\varepsilon_o \ .$$

The new motion parameter vector is given by

$$\vec{\gamma} = \vec{\gamma}_o + \vec{\delta},$$

and when the change in each component of the motion parameter vector becomes small, the algorithm is deemed to have converged and iteration stops. The covariance of the estimate $\vec{\gamma}$ is given by the following equation,

$$P = (X^T\sigma^{-1}X)^{-1}.$$

One of the key features of the MLP is the method employed to invert the symmetric, positive, semi-definite matrix, $X^T\sigma^{-1}X$. The usual inversion method is to use a Choleski decomposition, that is, to find an upper triangular matrix, $\bar{U}$, such that $\bar{U}^T\bar{U} = X^T\sigma^{-1}X$. The matrix $\bar{U}$ is easily inverted and $\bar{U}^{-1}\bar{U}^{-T}$ is the required inverse for finding the change in $\bar{\gamma}$. It would be desirable to find $\bar{U}$ directly and then the cross-product array would not have to be formed.

Such a procedure is possible if one uses a sequence of Givens' rotations (Reference 2) to decompose $\sigma^{-\frac{1}{2}}X$ into the product of an orthogonal matrix, Q, and an upper triangular matrix, $\bar{U}$. Then

$$X^T\sigma^{-1}X\delta = X^T\sigma^{-1}\varepsilon_o$$

becomes

$$\bar{U}^TQ^TQ\bar{U} = \bar{U}^TQ^T\sigma^{-\frac{1}{2}}\varepsilon_o.$$

Since Q is orthogonal, $Q^TQ = I$ and if $\bar{U}$ has an inverse, then the above becomes

$$\bar{U}\delta = Q^T\sigma^{-\frac{1}{2}}\varepsilon_o.$$

This triangular system of equations is easily solved, and the covariance matrix is

$$P = U^{-1}D^{-1}U^{-T} .$$

The Givens' notation is carried out on each row of X; hence, it is only necessary to store $\bar{U}$ and one row of X at a time. Figure 2.2a depicts the usual Givens' method approach. Note that M is the number of parameters estimated and N is the number of data points. Because this procedure is computationally expensive, there is an alternative procedure developed by Gentleman (Reference 3) which is used in the MLP algorithm. It is depicted in Figure 2.2b, and does not require any square roots and reduces the number of multiplications by one-quarter.

Thus, the revised algorithm calls for the accumulation of D, the diagonal matrix; $\bar{R}$, an upper unit triangular matrix; and θ, a vector of transformed residual errors. The array of weighted derivatives, $\sigma^{-\frac{1}{2}}X$, is replaced by $QD^{\frac{1}{2}}\bar{U}$, and $\sigma^{-\frac{1}{2}}\varepsilon_o$ is replaced by $D^{-\frac{1}{2}}Q^T\sigma^{-\frac{1}{2}}\varepsilon_o$, which is equal to θ. Using the normal equations, this yields:

$$X^T\sigma^{-1}X\delta = X^T\sigma^{-1}\varepsilon_o$$

$$\bar{U}^TD^{\frac{1}{2}}Q^TQD^{\frac{1}{2}}\bar{U} = \bar{U}^TD^{\frac{1}{2}}Q^T\sigma^{-\frac{1}{2}}\varepsilon_o$$

$$D\bar{U}\delta = D^{\frac{1}{2}}Q^T\sigma^{-\frac{1}{2}}\varepsilon_o$$

$$\bar{U}\delta = \theta .$$

CONSIDER TWO ROW VECTORS

$$0 \ldots 0 \quad \bar{U}_i \quad \bar{U}_{i+1} \ldots \bar{U}_k \ldots$$

$$0 \ldots 0 \quad x_i \quad x_{i+1} \ldots x_k \ldots$$

Transform by

$$\bar{U}'_k = C\bar{U}_k + Sx_k$$

$$x'_k = -S\bar{U}_k + Cx_k$$

where

$$\bar{U}'_i = \sqrt{\bar{U}_i^2 + x_i^2}$$

$$C = \bar{U}_i / \bar{U}'_i$$

$$S = x_i / \bar{U}'_i$$

To Yield

$$0 \ldots 0 \quad U'_i \quad U'_{i+1} \ldots U'_k \ldots$$

$$0 \ldots 0 \quad 0 \quad x'_{i+1} \ldots x'_k \ldots$$

THE PROCESS TRANSFORMS

$$\begin{bmatrix} X & X & X & X \\ & X & X & X \\ & & X & X \\ & & & X \\ X & X & X & X \end{bmatrix}$$

(New Row of X)

$\bar{U}$ ⟶ INTO ⟶

$$\begin{bmatrix} X & X & X & X \\ & X & X & X \\ & & X & X \\ & & & X \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$\bar{U}'$

REQUIRING ABOUT $2NM^2$ MULTIPLIES AND NM SQUARE ROOTS

FIG. 2.2a - GIVENS' ROTATIONS

CONSIDER $\bar{U} = D^{\frac{1}{2}}U$

D = Diagonal

R = Unit Upper Triangular Matrix

ROTATE

$$0 \ldots 0 \quad \sqrt{d} \quad \ldots \quad \sqrt{d}\, U_k$$

$$0 \ldots 0 \quad \sqrt{\delta} x_i \quad \ldots \quad \sqrt{\delta}\, x_k$$

Into

$$0 \ldots 0 \quad \sqrt{d'} \quad \ldots \quad \sqrt{d}\, U'_k$$

$$0 \ldots 0 \quad 0 \quad \ldots \quad \sqrt{\delta'} x'_k$$

By

$$d' = d + \delta x_i^2$$

$$\delta' = d\delta/d'$$

$$\bar{C} = d/d'$$

$$\bar{S} = \delta x_i/d'$$

$$x'_k = x_k - x_i U_k$$

$$\bar{r}'_k = \bar{C} r_k + \bar{S} x_k$$

REQUIRING ABOUT 3/2 NM^2 MULTIPLIES

FIG. 2.2b - SQUARE ROOT FORMULATION OF GIVENS' ROTATIONS

The reduction of the weighted residual errors to θ is accomplished by considering them as an augmenting column of the matrix X. The last equation involves a unit upper triangular matrix which makes the solution for δ particularly easy.

To determine which motion model to select, the MLP uses a statistical test, based on the residual errors generated by each model, called an F-test. Under the normal distribution of errors assumption, the residual sums of squares from each model are distributed as chi-square random variables and their ratios are distributed as F-random variables. Strictly, the F-test may only be applied when the errors are normally distributed, however, it has been shown to be a robust test against other distributions as long as they are symmetric about zero. The use of the ratio is valuable because it makes the test independent of scale changes in the input variance of the data.

In selecting a model it is desirable, intuitively, to favor the constant velocity model over the others because:

(1) It is the simplest of the four models.

(2) The other three models yield good solutions to a constant velocity data interval.

This is done by conducting a hypothesis test at a level, α. The test statistics are:

$$F_{ij} = (SS_i/DF_i)/(SS_j/DF_j)$$

$$i,j = 1,2,3,4;\ i<j$$

where SS_i and DF_i are the sum of squares of residuals and the degrees of freedom for the $i^{\underline{th}}$ model.

The hypothesis test is:

If $Pr(f>F_{1j}) \leq \alpha$

$j = 2,3,4$; accept Model 1

otherwise

If $Pr(f>F_{ij}) \leq .5$ for all $j>i$;

$i = 2,3,4$; accept Model i

It has been found that an α of about .2 works well. This value gives somewhat of an edge to Model 1 over Models 2, 3, and 4. The choice between Models 2, 3, and 4 is equal because the models are of roughly equal complexity.

An additional statistical test is performed to determine the proper data interval for the given motion model. Essentially, the test examines the residual stream from each data source for a mean shift, which is assumed to indicate that the target has undertaken some maneuver which cannot be described by the current motion model. When this shift is detected, a new data interval is begun and a new model is used.

The above exposition is not intended to be an exhaustive examination of the MLP algorithm. This is contained in Reference 4. Indeed, the MLP is significantly more complex and possesses far more detail than has been indicated here. For example, the MLP:

(1) Uses the Marquardt algorithm and several additional rules to speed convergence to the nonlinear solution;

(2) Provides and uses observability measures and accuracy measures for solutions generated by the algorithm;

(3) Provides for continuity constraints when switching from one motion model to another;

(4) Provides for the input of a priori state vector and covariance information;

(5) Provides an outlier accommodation scheme which has been incorporated into the measurement model.

Rather, the intent of this section is to give a brief overview of the main features of the MLP and to explain certain ideas which are used in both the MLP and hybrid algorithms.

2.2 Hybrid Algorithm

2.2.1 Introduction

The current Tracor hybrid algorithm is composed of two distinct entities--a scaled down MLP-type algorithm and

an extended Kalman filter--linked together by a set of statistical switching rules. The underlying idea of the algorithm is to use the batch filter to provide good initial estimates and, when needed, reinitialization, and to use the sequential filter to do the bulk of the tracking. The switching rules are intended, in the batch filter, to determine as quickly as possible when the filter has converged to a good solution and switch to the sequential, while in the sequential portion they are intended to determine when the sequential has lost track and switch to the batch for reinitialization. Studies performed at Tracor have shown that, in most cases, if a sequential filter of the type used in the hybrid is initialized with reasonably good starting values, it will be able to maintain even the most difficult track. Figure 2.2.1 contains a block diagram showing the relationship between the batch filter, the sequential filter, and the switching rules.

2.2.2 Aspects of Sequential Filter

Theory. The sequential filter used in the hybrid algorithm is a variant of the extended Kalman filter. The target's time rate of change of the acceleration vector is modeled as a Gaussian white noise process. This equation has the form

$$\frac{d\vec{a}}{dt} = \vec{n}$$

where $\vec{n}$ is Gaussian white noise with zero mean and covariance q_{xx} and q_{yy}. The solution, in the mean, to this stochastic differential equation is given by the following linear equations:

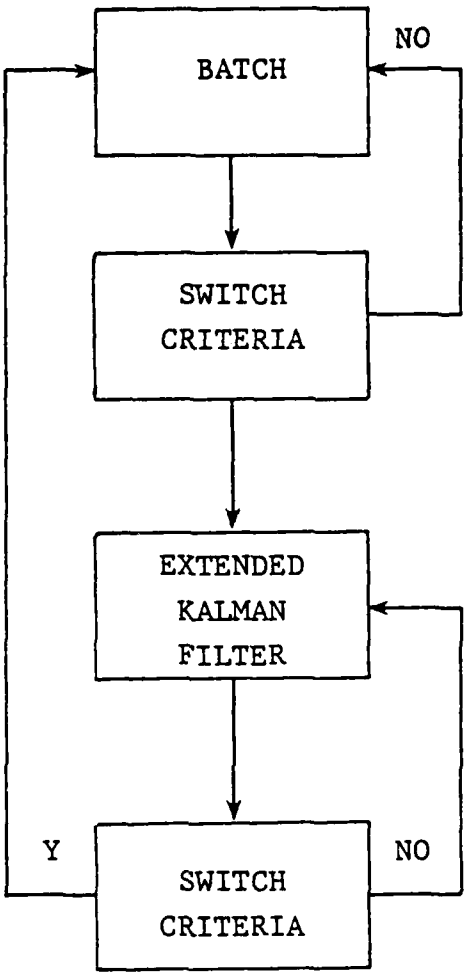


FIGURE 2.2.1 - HYBRID ALGORITHM MACRO STRUCTURE

$$\vec{r}(t) = \vec{r}(t_0) + \vec{v}(t_0)\Delta t + \tfrac{1}{2}\vec{a}(t_0)\Delta t^2$$

$$\vec{v}(t) = \vec{v}(t_0) + \vec{a}(t_0)\Delta t$$

$$\vec{a}(t) = \vec{a}(t_0)$$

where $\vec{r}$ and $\vec{v}$ are respectively the position and velocity vectors. The stochastic derivative of any tracking frequency is,

$$\frac{df}{dt} = \varepsilon$$

where ε is Gaussian white noise with zero mean and covariance q_{f_f}. The solution, in the mean, to this equation is

$$f = f_o.$$

Thus, at any time, t_k, the state vector for the model is:

$$X(t_k) = \begin{bmatrix} r_x(t_k) \\ r_y(t_k) \\ v_x(t_k) \\ v_y(t_k) \\ a_x(t_k) \\ a_y(t_k) \\ f_{o_1} \\ \vdots \\ f_{o_n} \end{bmatrix}$$

where r_x, r_y are the x and y position coordinates; v_x, v_y are the x and y velocities; a_x, a_y are the x and y accelerations; and $f_{o_1},\ldots,f_{o_n}$ are the different center frequencies being monitored. The state transition matrix for this system is

$$\Phi(t_k,t_{k-1}) = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \Delta t^2/2 & 0 & 0 \ldots 0 \\ 0 & 1 & 0 & \Delta t & 0 & \Delta t^2/2 & 0 \ldots 0 \\ 0 & 0 & 1 & 0 & \Delta t & 0 & 0 \ldots 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t & 0 \ldots 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \ldots 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \ldots 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \ldots 0 \\ & & & & \vdots & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \ldots 1 \end{bmatrix}$$

$$= \begin{bmatrix} R_{6\times 6} & O_{6\times n} \\ O_{n\times 6} & I_{n\times n} \end{bmatrix}$$

where $R_{6\times 6}$ is a matrix containing transition parameters for position, velocity, and acceleration only, $O_{6\times n}$ and $O_{n\times 6}$ are matrices of zeroes, and $I_{n\times n}$ is the identity matrix. Note that this corresponds to the constant acceleration motion model used in the MLP. However, the time periods over which the transition matrix is applied are far shorter than the time periods over which the MLP maneuver models are applied. This is one of the

major advantages of the sequential filter over the MLP. There is no need to test for appropriateness of model.

The measurement models are the same as those used in the MLP. Again, the primary models are Doppler-shifted frequency and bearing, with the capability to use range, Doppler ratio, Doppler difference, and time difference of arrival.

In order to obtain a practical propagation and update algorithm, the extended Kalman filter uses a linearized version of the measurement model, i.e., a first order Taylor series expanded about the most recent state estimate. For example, suppose a measurement of Doppler-shifted frequency, f_{r_j}, was received, then:

$$f_{r_j}(t_k, \vec{X}(t_k)) = f_{r_j}(t_{k-1}, \vec{X}(t_{k-1}))$$

$$+ \frac{\partial f_{r_j}}{\partial x}(t_{k-1}, \hat{\vec{X}}(t_{k-1})) \cdot (X(t_k) - \hat{\vec{X}}(t_{k-1}))$$

$$+ \frac{\partial f_{r_j}}{\partial y}(t_{k-1}, \hat{\vec{X}}(t_{k-1})) \cdot (Y(t_k) - \hat{Y}(t_{k-1})) + \ldots$$

$$+ \frac{\partial f_{r_j}}{\partial a_y}(t_{k-1}, \hat{\vec{X}}(t_{k-1})) \cdot (a_y(t_k) - \hat{a}_y(t_{k-1}))$$

$$+ \sum_{i=1}^{n} \delta_j^i \frac{\partial f_{r_j}}{\partial f_{o_i}}(t_{k-1}, \hat{\vec{X}}(t_{k-1})) \cdot (f_{o_i}(t_k) - \hat{f}_{o_i}(t_{k-1}))$$

where,

t_k = time of current estimate

$f_{r_j}(t_k, \vec{X}(t_k))$ = observed Doppler-shifted value of frequency r_j

t_{k-1} = time of last estimate

$\frac{\partial f_{r_j}}{\partial(\cdot)}(t_{k-1}, \hat{\vec{X}}(t_{k-1}))$ = partial derivative of received Doppler-shifted frequency f_{r_j} with respect to $(\cdot)$ evaluated at $t_{k-1}, \hat{\vec{X}}(t_{k-1})$

δ^i_j = Knonecker delta = $\begin{cases} 1 \text{ if } i=j \\ 0 \text{ if } i \neq j \end{cases}$

Bearing measurements can also be expanded in a first order Taylor series containing only position terms. Letting

$$d_k = f_{r_j}(t_k, \vec{X}(t_k)) - f_{r_j}(t_{k-1}, \hat{\vec{X}}(t_{k-1}))$$

we have, in vector form:

$$d_k = H_k(\vec{X}(t_k) - \hat{\vec{X}}(t_{k-1}))$$

where,

$$H_k = \left(\frac{\partial f_{rj}}{\partial x}\left(t_{k-1}, \hat{\vec{X}}(t_{k-1})\right), \ldots, \right.$$

$$\frac{\partial f_{rj}}{\partial a_y}\left(t_{k-1}, \hat{\vec{X}}(t_{k-1})\right), 0, \ldots,$$

$$\left.\frac{\partial f_{rj}}{\partial f_{oj}}\left(t_{k-1}, \hat{\vec{X}}(t_{k-1})\right), \ldots, 0\right)$$

$$\vec{X}(t_k) - \hat{\vec{X}}(t_{k-1}) = \left(x(t_k) - \hat{x}(t_{k-1}), \ldots, a(t_k) - \hat{a}(t_{k-1}), 0, \ldots,\right.$$

$$\left. f_{oj}(t_k) - \hat{f}_{oj}(t_{k-1}), \ldots 0\right)$$

The error covariance matrix, $P(t_k)$, for the parameters is given by:

$$P(t_k) = \phi(t_k, t_{k-1}) P(t_{k-1}) \phi^T(t_k, t_{k-1}) + \Gamma(t_k, t_{k-1}).$$

The matrix, $\Gamma(t_k, t_{k-1})$, contains the effects on the covariance of process noise and is found by:

$$\Gamma(t_k, t_{k-1}) = \int_{t_{k-1}}^{t_k} \phi(t, \tau) Q(\tau) \phi^T(t, \tau) d\tau.$$

This equation can be solved analytically because Q is assumed to be a constant matrix of the form:

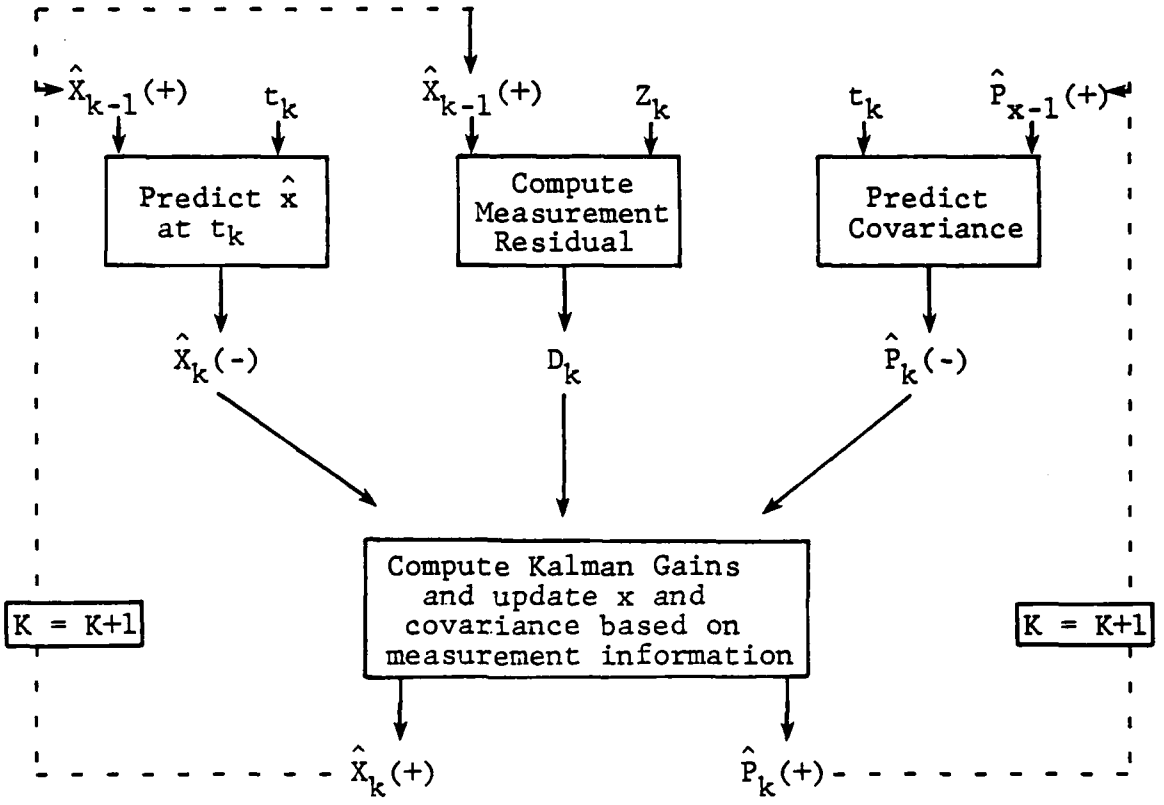
$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{a_x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{a_y} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_{f_{o_1}} & 0 \\ & & & & \vdots & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{f_{o_n}} \end{bmatrix}$$

Figure 2.2.2 contains a logical flow diagram of the sequential processor.

Implementation. There are several schemes which may be used to implement the sequential filter, each with its own set of advantages. The method of implementation chosen for the hybrid algorithm is the UDU^T square root form of the filter. For this particular problem it possesses several advantages:

(1) Helps prevent conditioning problems by insuring that the covariance matrix, P, will be positive definite and symmetric at all times.

(2) Helps insulate algorithm from effects of changes in computer word size, making it more suitable for operation on minicomputers than other forms of the algorithm.



$X_i(-)$ = Estimated state vector at time t_i before measurement

$X_i(+)$ = Estimated state vector at time t_i after measurement

t_i = Time t_i

$P_i(-)$ = Estimated covariance matrix at time t_i before measurement

$P_i(+)$ = Estimated covariance matrix at time t_i after measurement

Z_i = Measurement (Doppler shifted frequency, bearing, etc.) at time t_i

D_i = Measurement residual at time t_i

FIG. 2.2.2 - LOGIC DIAGRAM - KALMAN FILTER

(3) Minimizes storage requirements.

(4) Is compatible with Givens' rotation form of MLP.

A summary of the standard Kalman Filter conditions and equations is given in Figure 2.2.3. The following will describe how these equations are implemented under the UDU^T decomposition.

Recall that in Section 2.1.2 the covariance is defined as

$$P = U^{-1}D^{-1}U^{-T}$$

where U is upper triangular with ones on the diagonal and D is diagonal. This gives rise to the following definitions for the sequential:

$$P_k(-) = U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-)$$

$$P_k(+) = U_k^{-1}(+)D_k^{-1}(+)U_k^{-T}(+)$$

Note the following from Figure 2.2.3:

(1) $H_k(-)$ is a $1 \times n$ vector.

(2) R_k is a scalar.

(3) Q is constant for all t.

Substituting into the expression for the Kalman gain we get:

System Model:

$$X_k = \phi(t_k, t_{k-1})X_{k-1} + w(t) \qquad w(t) \sim N(0,Q)$$

Measurement Model:

$$Z_k = h_k\left(X(t_k)\right) + V_k \qquad V_k \sim N(0,R_k)$$

State Estimate Propagation:

$$\hat{X}_k(-) = \Phi(t_k, t_{k-1})\hat{X}_{k-1}(+)$$

Error Covariance Propagation:

$$P_k(-) = \phi(t_k, t_{k-1})P_{k-1}(+)\Phi^T(t_k, t_{k-1}) + \int_{t_{k-1}}^{t_k} \Phi(t,\tau)Q(\tau)\Phi^T(t,\tau)dt$$

Update estimates at time = t_k based on measurement Z_k.

State: $\hat{X}_k(+) = \hat{X}_k(-) + K_k[Z_k - h_k\left(\hat{X}_k(-)\right)]$

Covariance: $P_k(+) = P_k(-) - K_k H_k(-) P_k(-)$

Gain Matrix: $K_k = P_k(-)H_k^T(-)[H_k(-)P_k(-)H_k^T(-) + R_k]^{-1}$

where h_k = measurement model used at time k (Doppler shift, bearing, etc.)

$$H_k(-) = \frac{\partial h_k\left(X(t_k)\right)}{\partial X(t_k)} \qquad X(t_k) = \hat{X}_k(-)$$

FIG. 2.2.3 - KALMAN FILTER EQUATIONS

$$K_k = P_k(-)H_k^T(-)\left[H_k(-)P_k(-)H_k^T(-) + R_k\right]^{-1}$$

$$= U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-)H_k^{-T}(-)$$

$$\left[H_k(-)U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-)H_k^{-T}(-) + R_k\right]^{-1}$$

Let $V_k^T(-) = H_k(-)U_k^{-1}(-)$,

$$K_k = U_k^{-1}(-)D_k^{-1}(-)V_k^{-T}(-)\left[V_k^T(-)D_k^{-1}(-)V_k(-) + R_k\right]^{-1}$$

Since $V_k^T(-)$ is $1\times n$, $D_k(-)$ is $n\times n$, and R_k is a scalar, we have:

$$\alpha = V_k^{-T}(-)D_k^{-1}(-)V_k^{-1}(-) + R_k \qquad (\alpha = \text{a scalar}).$$

Thus,

$$K_k = \frac{U_k^{-1}(-)D_k^{-1}(-)V_k(-)}{\alpha}$$

Substituting this expression into the state estimate update gives:

$$\hat{X}_k(+) = \hat{X}_k(-) + \frac{U_k^{-1}(-)D_k^{-1}(-)V_k^{-T}(-)}{\alpha}\left(Z_k - H_k\left(\hat{X}_k(-)\right)\right) \qquad (1)$$

Substituting into the covariance update gives:

$$P_k(+) = P_k(-) - K_kH_k(-)P_k(-)$$

$$U_k^{-1}(+)D_k^{-1}(+)U_k^{-T}(+) = U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-)$$

$$- \frac{U_k^{-1}(-)D_k^{-1}(-)V_k(-)H_k(-)U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-)}{\alpha}$$

$$= U_k^{-1}(-)\left[D_k^{-1}(-) - \frac{D_k^{-1}(-)V_k^{-1}(-)V_k^{T}(-)D_k(-)}{\alpha}\right]U_k^{-T}(-) \quad (2)$$

For the error covariance propagation the decomposition gives:

$$P_k(-) = \Phi(t_k,t_{k-1})P_{k-1}(+)\Phi^T(t_k,t_{k-1}) + \Gamma(t_k,t_{k-1})$$

$$= \Phi(t_k,t_{k-1})U_{k-1}^{-1}(+)D_{k-1}^{-1}(+)U_{k-1}^{-T}(+)\Phi^T(t_k,t_{k-1}) + \Gamma(t_k,t_{k-1})$$

Let $S_{k-1}(+) = \Phi(t_k,t_{k-1})U_{k-1}^{-1}(+)$, then

$$P_k(-) = U_k^{-1}(-)D_k^{-1}(-)U_k^{-T}(-) = S_{k-1}(+)D_{k-1}^{-1}(+)S_{k-1}^{T}(+) + \Gamma(t_k,t_{k-1}) \quad (3)$$

Given the standard Kalman Filter equations expressed in square root format, the procedure is to solve equation (3) for $U_k(-)$ and $D_k(-)$. This is an easy and straightforward procedure to carry out since:

(1) $S_{ii} = 1$ because $\Phi_{ii}(t_k,t_{k-1}) = \mu k_{ii}(-) = \mu k_{ii}(+) = 1$; where μ_{ij} is an element in U^{-1}.

(2) The linear equations generated by equation (3) are all solvable by back substitution.

(3) For $j<i$, $\mu_{k_{ii}}(-) = 0$; for $i=j$, $\mu_{k_{ii}}(-) = \mu^T_{k_{ii}}(-) = 1$, allowing all the $d_{k_{ii}}(-)$ terms to be found. For $j>i$, $\mu_{k_{ii}}(-)$ can be found since all the $d_{k_{ii}}(-)$ terms required are known.

Once the $U_k^{-1}(-)$ and $D_k^{-1}(-)$ matrices have been determined, they can be substituted in equation (1) to give the updated estimate and substituted into equation (2) to give a set of equations to be solved for $U_k^{-1}(+)$ and $D_k^{-1}(+)$ that use a method identical to the one employed to get $U_k^{-1}(-)$ and $D_k^{-1}(-)$.

Other considerations in implementing the filter are:

(1) The matrix $\Phi(t_k,t_{k-1})$ is never stored. Whenever a multiplication is required the special nature of Φ is exploited.

(2) The $\Gamma(t_k,t_{k-1})$ integration is implemented in a closed form solution requiring no numerical integration techniques.

(3) Storage requirements are kept to a minimum. D_k^{-1} is stored as vector and U_k^{-1} is stored in row symmetric form without the diagonal elements.

2.2.3 Aspects of the Batch Filter

Theory. The batch portion of the hybrid algorithm is based on the assumption that, for short periods of time, the motion of a submarine can be approximated by the constant acceleration model of Section 2.2.2 (this model is quadratic in t, allowing it to fit tracks which are moderately curved). Once it has been determined that the batch algorithm has converged to a good solution, a switch is made to the sequential filter. Thus, in the hybrid algorithm, the batch filter's job is to converge, as quickly as possible, to a state vector which will allow the sequential filter to be properly initialized.

The measurement models used in the batch filter are the same as those used in the MLP and in the sequential filter, mainly bearing and Doppler-shifted frequency. As in the MLP, the batch filter uses data measurements to provide state vector estimates. The measurement models are approximated by first order Taylor series expansions. The design (X) matrix is made up of the corresponding partial derivatives, and the variation in the state vector is the quantity estimated (see Section 2.1.1). As in the MLP, a major share of the computational effort involved in the filter is in inverting the $X^{1}\sigma^{-1}X$ matrix, and, as in the MLP, this is done using a sequence of Givens' rotations (Section 2.1.1). In addition to the benefits described in Section 2.1.1, there is one further and very substantial benefit to using this method in the hybrid algorithm. Specifically, the U^{-1} matrix of the sequential is the inverse of the U matrix in the batch filter, and the D^{-1} matrix of the sequential is the inverse of the batch D matrix (Sections 2.1.1 and 2.2.1), i.e.,

$$U^{-1}_{(seq)} = U^{-1}_{(batch)}$$

$$D^{-1}_{(seq)} = D^{-1}_{(batch)}$$

Note that U and U^{-1} are unit upper triangular and that $D_{(seq)}$ and $D^{-1}_{(batch)}$ are diagonal, thus their inverses are easily found when switching from the batch to the sequential.

Since the batch filter contains only one motion model, there are no tests needed to select between various models and no tests needed to select a data interval over which a given model is appropriate. Since the motion model can be expressed in terms of the transition matrix used in the sequential filter, all forward and backward mappings can be performed with the same code used in the sequential filter. These two features reduce substantially the computer overhead imposed by the batch filter as compared with the overhead imposed by the MLP. Figure 2.2.4 contains a flowchart of the logic flow in the batch filter.

Implementation. As discussed above, several of the computational aspects of the batch filter are identical to those in either the sequential filter or the MLP. Briefly, these are:

(1) Solving the approximating linear least squares problem using Givens' rotations (MLP, Section 2.1.1).

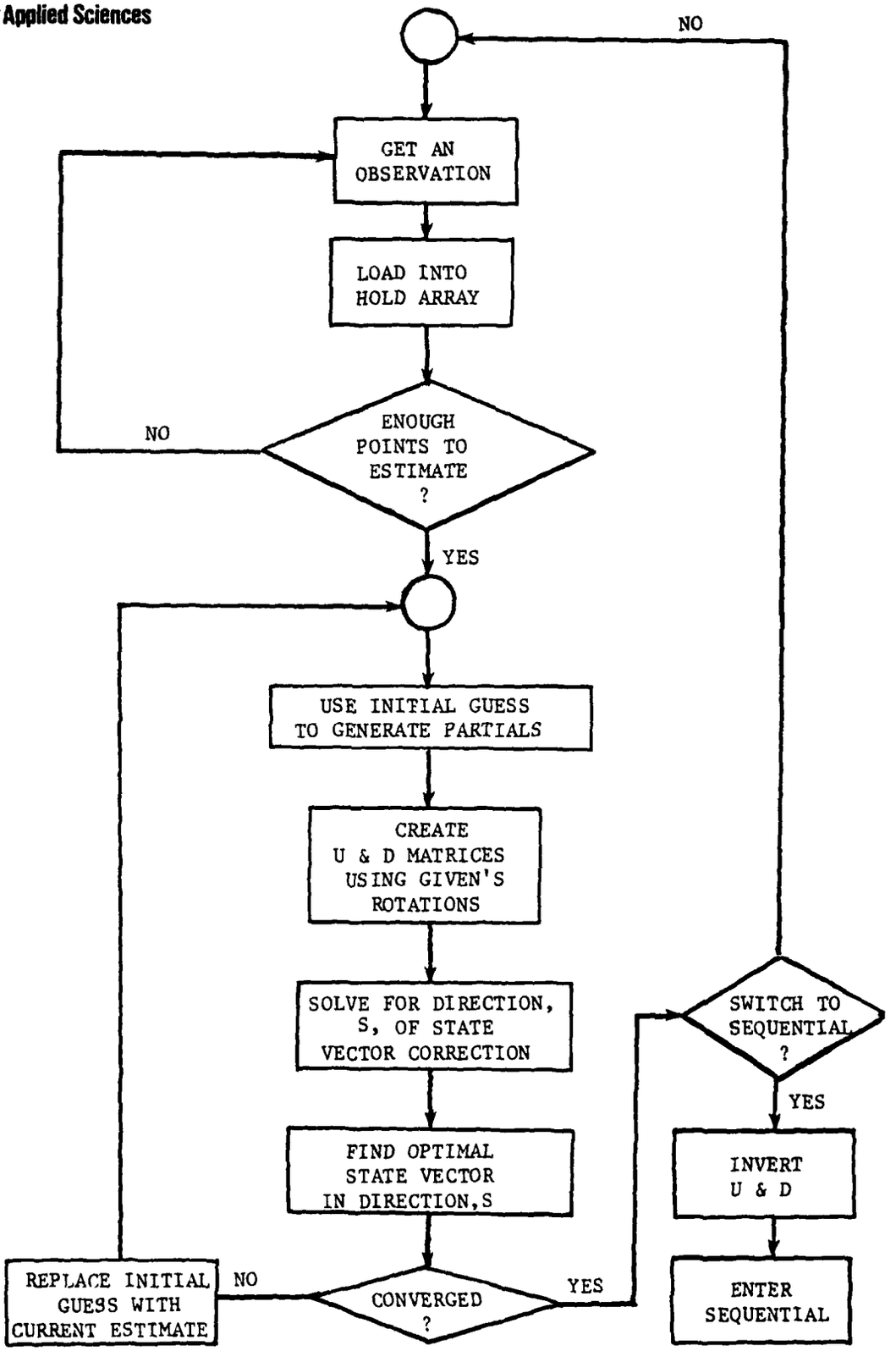


FIG. 2.2.4 - FLOWCHART - HYBRID BATCH INITIALIZER

(2) Projecting the track using a constant acceleration model (sequential, Section 2.2.1).

(3) Computing partial derivatives for a given measurement model (MLP, Section 2.1.1).

For a given number of data points, the batch filter generates a sequence of initial states, stopping when the sequence converges to a particular state vector. The sequence is said to have converged when:

$$\left| \frac{X_{(j)i} - X_{(j)i-1}}{X_{(j)i}} \right| \leq E \text{ for } j = 1,2,\ldots,n$$

where,

$X_{(j)i}$ = the $j^{\underline{th}}$ component of the state vector after the $i^{\underline{th}}$ iteration.

If the sequence hasn't converged after 30 iterations (i>j), a new point is read and the process is begun again.

One significant computational difference between the MLP and the batch filter is in minimizing the residual sum of squares once the variation in X has been found, using weighted least squares. The MLP uses the Marquardt (Reference 5) algorithm to speed convergence, while the batch filter uses a one-dimensional, quadratic search procedure. The algorithm is extremely fast and has several important properties:

(1) It searches in one direction only, avoiding the complications of multi-dimensional searches.

(2) For a quadratic function, the method converges quadratically to the minimum. Since even non-quadratic functions behave approximately quadratic in the region of a minimum, this assures rapid convergence in the final states of computation.

(3) It requires only information at the present stage and the one immediately previous to the present, thus reducing substantially the amount of core and computation required.

(4) The method requires no derivatives.

2.2.4 Switching Rules

2.2.4.1 Batch to Sequential. In the hybrid algorithm the basic assumptions are that a simplified MLP algorithm can be used to fit the data, the fit can be detected early, and a switch can be made to a sequential filter using the initial conditions supplied by the batch filter. Thus, the batch-to-sequential switching must try to satisfy two constraints:

(1) The transition from the batch filter to the sequential filter should be made as quickly as possible.

(2) The switch from batch to sequential should not be made until a good approximation to the initial state vector has been found.

To resolve these problems the hybrid algorithm uses a nonlinear test of significance developed by Gallant (Reference 6). The Gallant method is used to test the hypothesis

$$H_0: \vec{X}_i = \vec{X}_j \text{ versus } H_1: \vec{X}_i \neq \vec{X}_j$$

where $\vec{X}_i$ is the estimated state vector from the first i data points and X_j is the estimated state vector from the first j data points. Currently, i = j + 2*NBY, where NBY is the number of buoys in the scenario. Gallant's test statistic for this hypothesis is:

$$T_{(i,j)} = \frac{\sum_{k=1}^{i} V_k \cdot \left(Y_k - \hat{Y}_k(\vec{X}_j)\right)^2}{\sum_{k=1}^{i} V_k \cdot \left(Y_k - \hat{Y}_k(\vec{X}_i)\right)^2}$$

where

Y_k = $k^{\underline{th}}$ observed measurement value

$\hat{Y}_k(\vec{X}_p)$ = predicted $k^{\underline{th}}$ measurement value based on estimated state vector $\vec{X}_p(p=i,j)$

V_k = weight associated with $k^{\underline{th}}$ measurement

The critical value for the test is:

$$C(i) = 1. + \frac{pF_{\alpha}(p,i-p)}{i-p}$$

where

p = number of elements in the state vector

i = number of points used to estimate state vector X_i

$F_{\alpha}(p,i-p)$ = α percentage point of F-distribution.

Lastly, the decision rule is:

(1) If T is greater than C, reject H_0.

(2) If T is less than C, accept H_0.

Basically, the idea is to compare the weighted residual sum of squares for estimates. If the ratio is too large, we reject the idea that the estimates are equal, otherwise, we accept it. Gallant's critical point gives us a convenient and consistent measure of "too large".

If $i = j + 2 \cdot n$ and $k = i + 2 \cdot n$, where n is the number of buoys, then a switch is made from the batch to the sequential when

$$T_{(i,j)} \leq C_{(i)} \text{ and } T_{(k,i)} \leq C_{(k)} .$$

Figure 2.2.4.1 is a flowchart of the batch-to-sequential switching logic. There are two primary reasons for requiring the data to satisfy the switching rules two times in succession:

(1) By chance, a certain percentage of the time the data will give a false reading due to the inherent random nature of the data.

(2) During the earliest stages of estimation estimates of the state vector may be quite poor and may not change much over 2•n points. Thus, based on the data, the test may conclude that the batch filter has converged when, in fact, it hasn't. Usually another run through 2•n points is enough to determine whether or not the process has converged.

The basic idea behind this approach is to use the data itself to determine when the batch filter has converged. At present an alpha level of .005 is being used, however, this could become an operator input if desired.

2.2.4.2 Sequential to Batch. The sequential filter used in the hybrid algorithm is capable of maintaining track under a wide range of conditions. However, studies at Tracor have shown that there are times when it will lose track, for example, if there is a long period of time when data is received from only one sensor due to low signal-to-noise ratios. Thus, the job of the sequential-to-batch switching rules is to sense when the sequential filter has lost track and cause a transfer

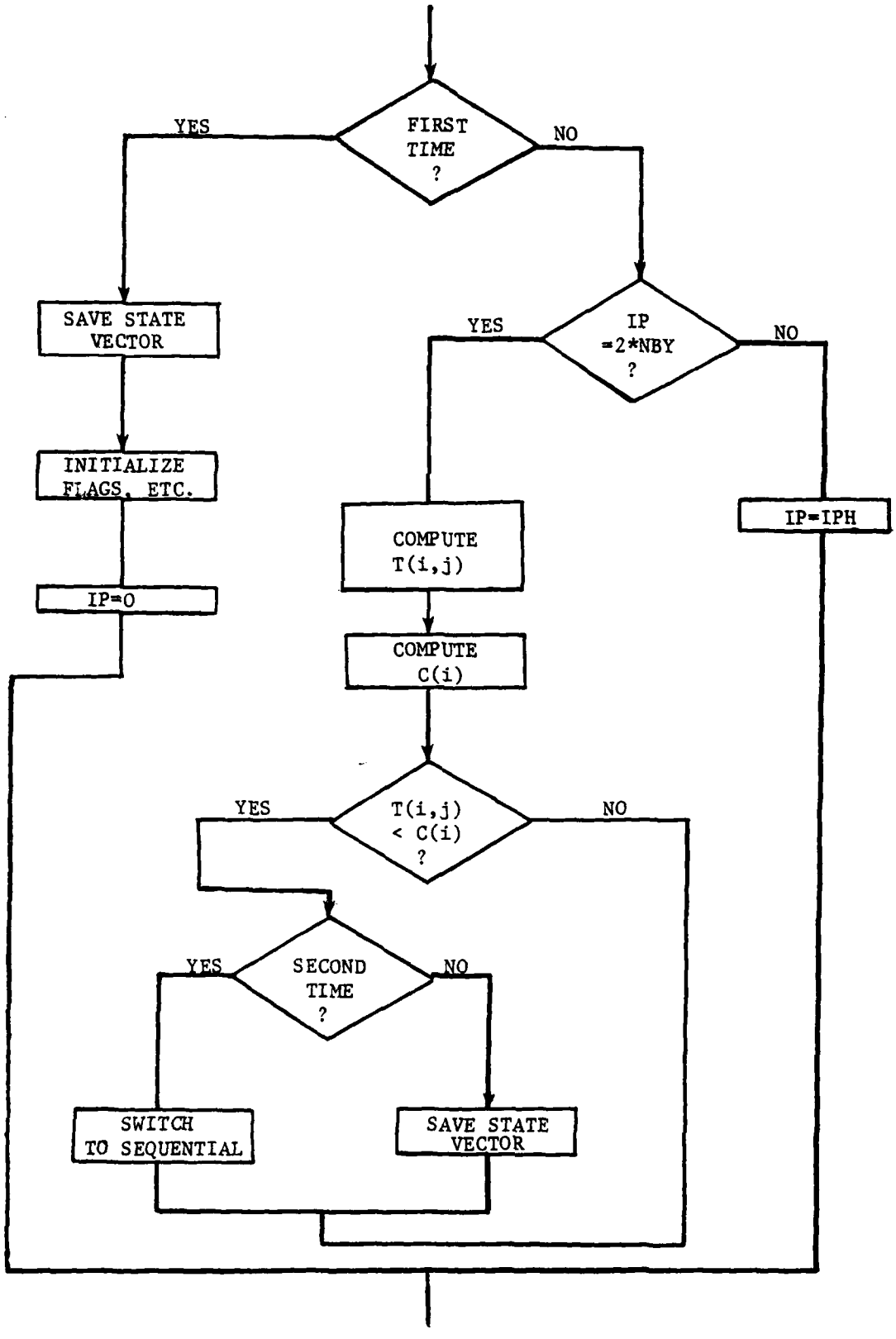


FIG. 2.2.4.1 - FLOWCHART - BATCH SWITCHING LOGIC

back to the batch filter for reinitialization. Note, an implied constraint is that the procedure must not be so sensitive as to interpret normal maneuvers as losing track.

The hybrid contains two tests to determine when the sequential filter has lost track. The first is a gross test, and failure to pass it will cause an immediate switch to the batch filter. Essentially, it consists of a comparison of the residuals obtained before and after the filtering. If the residual after correction is larger than the residual before correction, the filter is assumed to be diverging and a switch is made to the batch. This is a fairly insensitive test; however, it is fast, easy to do, and provides "worst case" protection.

The second test has proved to be quite successful in determining loss of track. It is based on two aspects of the Kalman Filter:

(1) If state vector estimate $\hat{X}_{k-1}$ is unbiased, then state vector estimate $\hat{X}_k$ is unbiased (Reference 7), i.e., $E[\hat{\vec{X}}_k] = E[\vec{X}_k]$.

(2) The errors associated with the measurement model are assumed to be normally distributed with zero mean.

Define the $i^{\underline{th}}$ measurement residual to be

$$Y_i - \hat{Y}_i(\hat{\vec{X}}_i)$$

where

Y_i = $i^{\underline{th}}$ observed measurement value

$\hat{Y}_i(\hat{\vec{X}}_i)$ = $i^{\underline{th}}$ predicted measurement value based on estimated state vector $\hat{\vec{X}}_i$

The residuals are the differences between what was observed at the $i^{\underline{th}}$ data point and what the filter predicted at that point, and may be thought of as the observed errors if $\hat{\vec{X}}_k$ is near $\vec{X}_k$. Under aspect (2) above, the residuals should be distributed as normal random variables with zero mean if, again, $\hat{\vec{X}}_k$ is near $\vec{X}_k$. Specifically, if the residuals are plotted over time and exhibit no trends, such as ramping or dramatic mean shifts, then $\hat{\vec{X}}_k$ is probably near $\vec{X}_k$; at least the data does not violate this assumption.

Thus, the problem of sequential-to-batch switching may be viewed as a problem in determining whether or not a trend exists in the measurement residuals. To do this the hybrid fits a regression line of the form,

$$\hat{y} = \hat{b}_0 + \hat{b}_1 t \quad (t = \text{time})$$

to the residuals from each sensor for each data type. The lines are tested for significance using an F-test (currently α equals .005) and, if any one of them tests as significant for four consecutive data points a switch is made to the batch filter.

Note that a separate line is computed for each data type from each sensor. This is done to prevent information from one or two sensors, which may be sensitive to the filter

losing track, from being buried or obscured by the information from several other sensors which are insensitive to the filter losing track due to scenario geometry, poor signal-to-noise ratios, etc. There are several reasons for requiring the line to test significantly for more than one data point:

(1) Early in the data stream it is very possible that the residuals will cause a significant fit to occur when, in fact, track has not been lost.

(2) The sequential filter will typically wander off, at different times, for one or two points. This allows it time to regain track on its own.

(3) One or two outliers at any point in the data stream could cause an indication of significance when, in fact, there is none.

(4) It prevents sharp or rapid maneuvers from being interpreted as lost track.

To allow the hybrid to be altered to fit various environmental conditions, it would be quite easy to allow the F-test significance level and the number of required significant points to be operator inputs. Then, for example, if the environment was noisy with low data rate the alpha level could be reduced and the number of significance points could also be reduced, causing the sequential to be more sensitive to any loss of track.

2.3 Iterated Sequential Algorithm

2.3.1 Aspects of Iterated Sequential Algorithm

The iterated sequential algorithm, in form, is very similar to the hybrid algorithm. That is, both algorithms are comprised of three primary elements--an initializer, a sequential Kalman Filter, and a set of switching rules for moving from the initializer to the tracker and back again. The only difference in the two procedures is in the initializer. The hybrid uses a batch algorithm to initialize, while the iterated sequential uses the same filter to initialize that it does to track, relying on iteration to produce convergence. There were several reasons for implementing this algorithm:

(1) There is a reduction of core requirements. Instruction requirements are reduced because the same subroutines used in the straight sequential filter are used in the initializer. Data requirements are reduced because fewer save arrays are needed to make the transition between initializer to tracker.

(2) There is faster execution time. The iterated sequential starter uses fewer operations to achieve an estimate than does the batch, however, iterating about an answer could negate this to some degree.

(3) One possible weakness of the hybrid is attempted initialization during a track too difficult to follow with the constant acceleration model. By its sequential structure the iterated starter should be able to avoid this problem.

(4) This algorithm has simpler program organization. In the hybrid, provisions must be made to accommodate two distinctly different algorithms and transfers between them. In the iterated sequential, the same operations are performed in both portions of the algorithm, simplifying considerably storage management and control transfer between the initializer and tracker.

Figure 2.2.1 can equally well represent the iterated sequential algorithm as the batch; simply replace "BATCH" with "ITERATED SEQUENTIAL". The extended Kalman Filter used for tracking is identical to that used in the hybrid (see Section 2.2.2) and the switching rules (Section 2.2.4) are also the same.

Figure 2.3.1 contains a logical flowchart of the iterated sequential starter. The basic idea is to use accumulated data points to project initial state vector and covariance estimates forward, using the sequential filter, and then test for convergence. Convergence occurs when

$$\left| \frac{X_{(j)i} - X_{(j)i-1}}{X_{(j)i}} \right| \leq E \text{ for } j = 1,2,3,\ldots,n$$

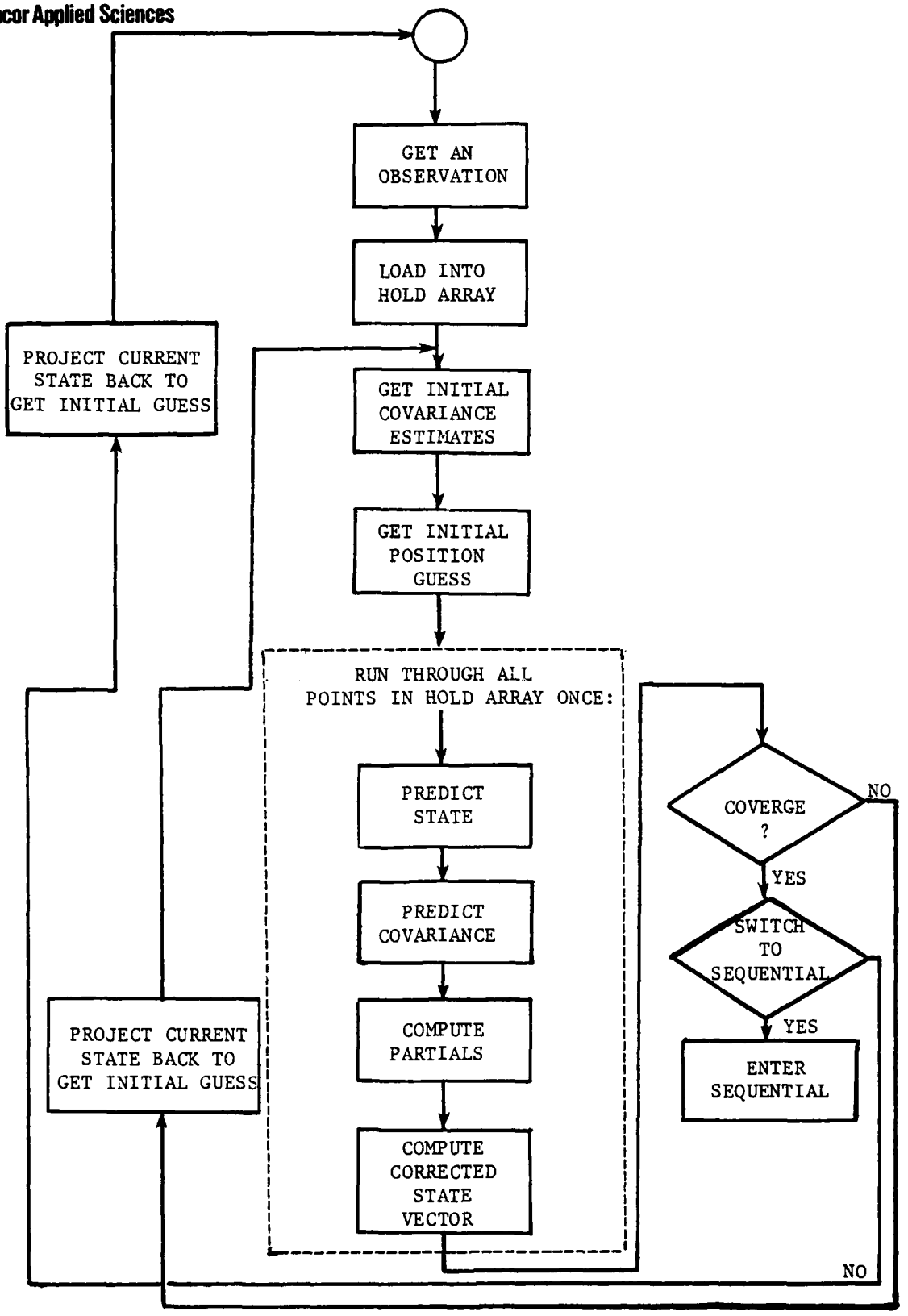


FIG. 2.3.1 - FLOWCHART - ITERATED SEQUENTIAL INITIALIZER

where,

$X_{(j)i}$ = the $j^{\underline{th}}$ component of the state vector after the $i^{\underline{th}}$ iteration

If the convergence does not occur on the $i^{\underline{th}}$ iteration, the state vector is projected back to time, t_o, using the constant acceleration model (Sections 2.1 and 2.2.2) as implemented in the sequential filter. When convergence does occur, a switch test is made. If the decision is made to switch, the straight sequential filter is entered; if not, the state is projected back (as above), a new data point is read, and the cycle begins anew.

3.0 TEST RESULTS (COMPARISONS)

3.1 Introduction

This section presents the results of several tests designed to measure the relative tracking abilities of each of the three algorithms examined in this report. The tests consisted of four different scenarios, each replicated twenty-five times as a series of Monte Carlo runs. The algorithms then processed each scenario and the results were analyzed according to four criteria. These requirements are:

(1) Average distance error, $\bar{\delta}$, throughout the track, defined as,

$$\bar{\delta} = \int_{t_i}^{t_f} \frac{\delta(t)dt}{t_f - t_i}$$

where,

$$\delta(t) = \vec{r}_E(t) - \vec{r}_T(t)$$

$\vec{r}_E(t)$ = estimated position vector at time, t

$\vec{r}_T(t)$ = true position vector at time, t

This is a measure of the average closeness of the true and estimated trajectories for the time interval (t_i, t_f). The two trajectories are considered close when

$$\bar{\delta} < \delta^*$$

For some specified level of δ^* (currently 500 meters).

(2) Percent holding time, $\bar{\tau}$, throughout the track, defined as,

$$\bar{\tau} = \frac{\sum_{j=1}^{n} (\tau f_j - \tau i_j)}{t_f - t_i} \cdot 100\%$$

where, for a given time interval (t_i, t_f),

$$(\tau i_j, \tau f_j) \subseteq (t_i, t_f)$$

$$(\tau i_j, \tau f_j) \cap (\tau i_k, \tau f_k) = \phi \text{ for } j \neq k$$

$$\bar{\delta}_j = \int_{i_j}^{f_j} \frac{\delta(t)dt}{\tau f_j - \tau i_j} \leq \delta^* \text{ for } j = 1,2,\ldots,n.$$

(3) Predictive ability. Assuming the target stays on the same trajectory, a given algorithm will be said to have good predictive ability if

$$\delta(t_f + \Delta t) \leq \delta^*.$$

(4) Average CPU time used to process one scenario. This gives a measure of the relative speeds of the three algorithms.

Note, again, that these measures are computed over twenty-five Monte Carlo samples and are intended as a measure of the average

or expected tracking abilities of the algorithms.

3.2 Scenario Descriptions

In testing the MLP, hybrid, and iterated sequential algorithms the intent was to produce several scenarios, covering a range of tracking difficulties, to test the capabilities of the three algorithms. The intent was not, at this time, to evaluate each algorithm at the extreme edge of its performance range.

The basic layout for each of the four scenarios generated was the same. It consisted of three DIFAR buoys arranged in the shape of an equilateral triangle with 2,000 meter sides; no buoy drift was assumed. All maneuvers were assumed to take place within, or near, the buoy field and a sea state of two was assumed. Figure 3.2e summarizes the environmental conditions and inputs common to all four scenarios.

The first scenario run (Figure 3.2a) was a simple straight line course through the center of the buoy field. The intent of this scenario was to determine whether or not the three algorithms could track straight line motion with high efficiency.

The second scenario considered (Figure 3.2b) was a 180 degree turn in the middle of the buoy field. This was felt to be a scenario of moderate difficulty and was intended to compare the ability of each algorithm to respond to an extremely sharp turn. Note that the maneuver time for this turn is about 300 seconds. This not only includes the time required to make the turn itself, but also the time needed to

regain the required operating velocity (in this case 7 meters/second).

Scenario 3 (Figure 3.2c) was considered to be of high difficulty and consisted of a 360 degree turn performed inside the buoy field. This was intended to test the ability of each algorithm to follow a severe maneuver for a prolonged length of time. Again, the maneuver time indicated in the figure included not only the time required to make the circle, but also the time required to regain an operating velocity of 7 meters/second.

The final scenario (Figure 3.2d) was also considered to be of severe difficulty and was intended to measure the ability of each algorithm to follow a sequence of fairly rapid turns and also to compare the ability of each algorithm to follow maneuvers outside, but near, the buoy field.

3.3 Generation of Data for DIFAR Simulation

In order to carry out simulations of the three target tracking algorithms examined in this report, an algorithm for generating realistic data, as it would appear from a general processor for DIFAR buoys, was developed. It was desired that the generated data $\{(f_i,\beta_i,t_i)\}$ have a statistical distribution closely approximating real data.

3.3.1 General Approach

The general processor modeled (see Figure 3.3.2.1) consists of a filter band with bins of width Δf, followed by a square law detector [$(\ )^2$ and a $1/\Delta f$ averager] and a finite

Duration: 630 seconds

Initial Conditions:

Position			Velocity	Course
X	Y	Z		
-1200.0	0.0	50.0	7.0 m/s	53.0 deg.

Maneuvers: None

Final Conditions:

Position			Velocity	Course
X	Y	Z		
1453.2	3521.6	50.0	7.0 m/s	53.0 deg.

Maneuver Description: Straight line track through center of buoy field.

FIG. 3.2a - SCENARIO 1 DESCRIPTION

Duration: 770 seconds

Initial Conditions:

Position			Velocity	Course
X	Y	Z		
-1200.0	0.0	50.0	7.0 m/s	53.0 deg.

Maneuvers:

Time		Position X	Position Y	Position Z	Velocity	Course
Begin	285.0	0.6	1593.3	50.0	7.0 m/s	53.0 deg.
Complete	595.0	-765.1	899.6	50.0	7.0 m/s	240.0 deg.

Final Conditions:

Position			Velocity	Course
X	Y	Z		
-1377.4	-160.9	50.0	7.0 m/s	240.0 deg.

Maneuver Description: Hairpin turn in center of buoy field.

FIG. 3.2b - SCENARIO 2 DESCRIPTION

Duration: 765 seconds

Initial Conditions:

Position			Velocity	Course
X	Y	Z		
-1200.0	0.0	50.0	7.0 m/s	53.0 deg.

Maneuvers:

Time		Position			Velocity	Course
		X	Y	Z		
Begin	285.0	.6	1593.3	50.0	7.0 m/s	53.0 deg.
Complete	720.0	732.2	2611.8	50.0	7.0 m/s	50.0 deg.

Final Conditions:

Position			Velocity	Course
X	Y	Z		
934.7	2853.1	50.0	7.0 m/s	50.0 deg.

Maneuver Description: Small closed loop inside buoy field.

FIG. 3.2c - SCENARIO 3 DESCRIPTION

Duration: 90 seconds

Initial Conditions:

Position			Velocity	Course
X	Y	Z		
-1200.0	0.0	50.0	7.0 m/s	82.0 deg.

Maneuvers:

Time		Position X	Y	Z	Velocity	Course
Begin	125.0	-1078.2	866.5	50.0	7.0 m/s	82.0 deg.
Complete	240.0	-1402.9	1549.1	50.0	7.0 m/s	121.0 deg.
Begin	255.0	-1457.0	1639.1	50.0	7.0 m/s	121.0 deg.
Complete	325.0	-1343.0	1987.5	50.0	7.0 m/s	20.0 deg.

Final Conditions:

Position			Velocity	Course
X	Y	Z		
2899.0	3531.7	50.0	7.0 m/s	50.0 deg.

Maneuver Description: Right-left turn combination just below buoy field.

FIG. 3.2d - SCENARIO 4 DESCRIPTION

Signal Level: 140 dB

Ambient Noise Level: 67 dB

Number of Observing Buoys: 3

Buoy Positions:

Buoy	X	Y	Z
1	-1000.0	1000.0	50.0
2	1000.0	1000.0	50.0
3	0.0	2732.0	50.0

Initial State Vector Guess:

$\underline{X}$	$\underline{Y}$	$\dot{\underline{X}}$	$\dot{\underline{Y}}$	$\ddot{\underline{X}}$	$\ddot{\underline{Y}}$	$\underline{f_o}$
0.0	1500.0	4.0	4.0	0.0	0.0	150.0

Initial Covariance Matrix:

$$\begin{bmatrix} 10{,}000.0 & & & & & & 0 \\ & 10{,}000.0 & & & & & \\ & & 1.0 & & & & \\ & & & 1.0 & & & \\ & & & & .1 & & \\ & & & & & .1 & \\ 0 & & & & & & .1 \end{bmatrix}$$

Data Types Processed: Frequency, Bearing

Data Rate: 1 OBS / 20 Seconds

FIG. 3.2e - COMMON SCENARIO INPUT AND ENVIRONMENTAL VARIABLES

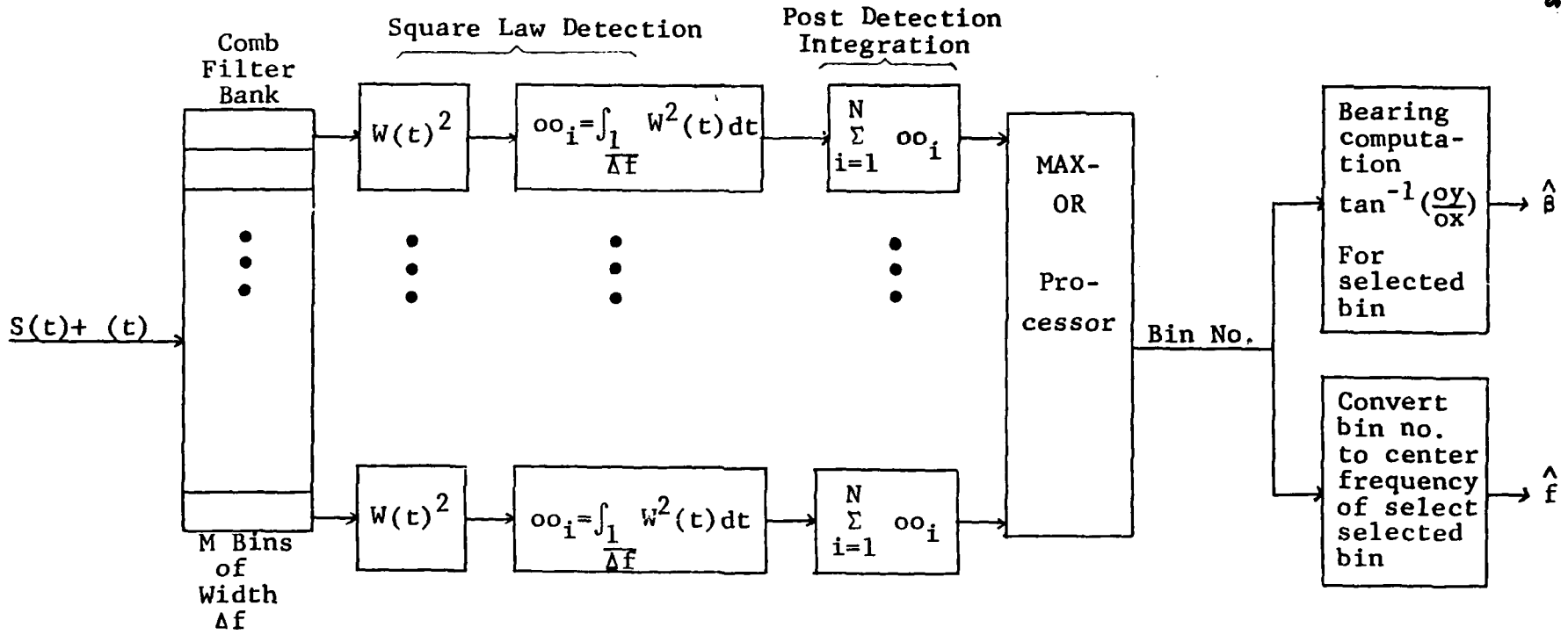


FIGURE 3.3.2.1 - GENERAL PROCESSOR FOR DIFAR BUOY

time perfect integrator (ALI). A MAX-OR picks the strongest signal in frequency from the omni signal channel and computes bearing from the X and Y channels. The following describes the statistics of the outputs of various components in such a processor. The distribution of the envelope of a Gaussian process: (1) at the output of a narrowband filter is Rayleigh distributed, (2) at the output of a narrowband filter and square law detector is exponential, and (3) at the output of a narrowband filter, square law detector, post detection integrator is CHI-Squared distributed. One could randomly generate errors from these distributions to add to the correct data; however, with signal present, the X and Y signals are not independent.

3.3.2 Analysis

3.3.2.1 Generation of Frequency Data. It is assumed that the input to the processor may be written as the sum of a narrowband Gaussian signal and Gaussian noise,

$$W(t) = S(t) + n(t)$$

where

$$S(t) \sim N(0,\sigma_S^2)$$

$$n(t) \sim N(0,\sigma_n^2).$$

At the output of a narrowband filter, the signal term may be represented as the sum of two orthogonal signals, each multiplied by an independent Gaussian random variable.

$$S(t) = S\cos\omega_o t + S_2\sin\omega_o t$$

where

$$S_1 \sim N(0,\sigma_S^2)$$

$$S_2 \sim N(0,\sigma_S^2)$$

ω_o = center frequency of filter

Similarly for the noise term,

$$\eta(t) = \eta_1\cos\omega_o t + \eta_2\sin\omega_o t$$

where

$$\eta_1 \sim N(0,\sigma_\eta^2)$$

$$\eta_2 \sim N(0,\sigma_\eta^2).$$

In the processor, W(t) is squared and time averaged over $1/\Delta f$ seconds, where Δf is the filter bandwidth in Hertz.

$$\overline{W(t)^2} = \Delta f \int_0^{1/\Delta f} [S_1^2\cos^2\omega_o t + S_2^2\sin^2\omega_o t + \eta_1^2\sin^2\omega_o t + \eta_2^2\cos^2\omega_o t + 2S_1\eta_1\cos^2\omega_o t + 2S_2\eta_2\cos^2\omega_o t + 2S_1S_2\cos\omega_o t\sin\omega_o t + 2\eta_1\eta_2\cos\omega_o t\sin\omega_o t + 2S_1\eta_2\cos\omega_o t\sin\omega_o t + 2S_2\eta_1\cos\omega_o t\sin\omega_o t]dt.$$

It is assumed that S_1, S_2, η_1, and η_2 are slowly varying compared to $\cos^2\omega_o t$, $\sin^2\omega_o t$, and $\cos\omega_o t\sin\omega_o t$. These may then be pulled outside the integral sign. The terms with $\cos\omega_o t\sin\omega_o t$ factors integrate to zero.

$$\overline{W(t)^2} = \Delta f[(S_1^2 + \eta_2^2 + 2S_1\eta_1)\int_0^{1/\Delta f}\cos^2\omega_o t\,dt + (S_2^2 + \eta_1^2 + 2S_2\eta_2)\int_0^{1/\Delta f}\sin^2\omega_o t\,dt].$$

Integration yields

$$\overline{W(t)^2} = \Delta f[(S_1^2 + \eta_2^2 + 2S_1\eta_1)\ (\tfrac{1}{2}t + \frac{1}{4\omega_o}\sin 2\omega_o t) + (S_2^2 + \eta_1^2 + 2S_2\eta_2)\ (\tfrac{1}{2}t - \frac{1}{4\omega_o}\sin 2\omega_o t)]\Big|_0^{1/\Delta f}$$

Since the process is narrowband, $\Delta f << \omega_o$

$$W(t)^2 = \tfrac{1}{2}(S_1^2 + \eta_2^2 + 2S_1\eta_1) + \tfrac{1}{2}(S_2^2 + \eta_1^2 + 2S_2\eta_2).$$

Using the transformations

$$S_1 = \sigma_S N_1$$

$$S_2 = \sigma_S N_2$$

$$n_1 = \sigma_\eta N_3$$

$$n_2 = \sigma_\eta N_4$$

where N_1, N_2, N_3, $N_4 \sim N(0,1)$, the expression becomes

$$W(t)^2 = \frac{\sigma_S^2}{2}(N_1^2 + N_2^2) + \frac{\sigma_\eta^2}{2}(N_3^2 + N_4^2) + \sigma_S\sigma_\eta(N_1N_3 + N_2N_4).$$

Since the quantity of interest is the power relative to the noise power σ_η^2, the equation is divided through by σ_η^2. Letting

$$\rho = \frac{\sigma_S^2}{\sigma_\eta^2},$$

the signal-to-noise ratio, the relative power is

$$OO = \frac{W(t)^2}{\sigma_\eta^2} = \tfrac{1}{2}\rho(N_1^2 + N_2^2) + \tfrac{1}{2}(N_3^2 + N_4^2) + \sqrt{\rho}(N_1N_3 + N_2N_4).$$

Since the signal may be smeared over several frequency bins, it is necessary to weight the signal-to-noise ratio by γ_i, the fraction of the integration time that the target spends in bin i. The expression for OO becomes

$$OO_i = \tfrac{1}{2}\gamma_i\rho(N_1^2 + N_2^2) + \tfrac{1}{2}(N_3^2 + N_4^2)\ \sqrt{\gamma_i\rho}(N_1N_3 + N_2N_4).$$

It is seen that the average power relative to noise power out of the narrowband filter of an omnidirectional hydrophone can be simulated by knowing the signal-to-noise ratio and generating four independent samples from a standard normal distribution. The bin with the largest value for OO is chosen by the MAX-OR processor and the center frequency of that bin is used as the frequency data point.

3.3.2.1 Generation of Bearing Data. To arrive at an estimate of bearing, the voltage from the omniphone is multiplied and time averaged with the voltage from the X phone and the voltage from the Y phone. The bearing estimate β is then

$$\beta = \tan^{-1}\left(\frac{OY}{OX}\right)$$

where OY and OX are the averages mentioned above.

The equations used in the simulation for OX shall be derived. The derivation of OY is similar. The input to the omniphone is

$$W(t) = S(t) + n(t)$$

as before. The input to the X phone is

$$V(t) = S(t)\cos(\theta) + n_x(t)$$

where θ is the true target bearing and

$$n_x(t) \sim N\left(0, \frac{\sigma_n^2}{2}\right).$$

It can be shown that $\eta(t)$ and $\eta_x(t)$ are independent processes. In terms of orthogonal components,

$$W(t) = S_1 \cos\omega_o t + S_2 \sin\omega_o t + \eta_1 \cos\omega_o t + \eta_2 \sin\omega_o t \ ;$$

$$V(t) = S_1 \cos\omega_o t \cos\theta + S_2 \sin\omega_o t \cos\theta + \eta_{x_1} \cos\omega_o t + \eta_{x_2} \sin\omega_o t$$

where

$$S_1 \sim N(0, \sigma_S^2)$$

$$S_2 \sim N(0, \sigma_S^2)$$

$$\eta_1 \sim N(0, \sigma_\eta^2)$$

$$\eta_2 \sim N(0, \sigma_\eta^2)$$

$$\eta_{x_1} \sim N(0, \frac{\sigma_\eta^2}{2})$$

$$\eta_{x_2} \sim N(0, \frac{\sigma_\eta^2}{2}).$$

$W(t)$ and $V(t)$ are multipled and time averaged. It is assumed that S_1, S_2, η_1, η_2, η_{x_1}, and η_{x_2} are slowly varying. The terms with $\cos\omega_o t \sin\omega_o t$ factors integrate to zero. The expression is then

$$\overline{W(t)\ V(t)} = \int_o^{\frac{1}{\Delta f}} [\cos^2\omega_o t (S_1^2 \cos\theta + \eta_1 \eta_{x_1} + S_1 \eta_{x_1} + \eta_1 S_1 \cos\theta)$$

$$+ \sin^2\omega_o t (S_2^2 \cos\theta + \eta_2 \eta_{x_2} + S_2 \eta_{x_2} + \eta_2 S_2 \cos\theta)] \ dt.$$

Integration yields

$$\overline{W(t)\ V(t)} = \tfrac{1}{2}(S_1^2\cos\theta + \eta_1\eta_{x_1} + S_1\eta_{x_1} + \eta_1 S_1\cos\theta)$$

$$+ \tfrac{1}{2}(S_2^2\cos\theta + \eta_2\eta_{x_2} + S_2\eta_{x_2} + \eta_2 S_2\cos\theta).$$

Using the transformations and the following transformations,

$$\eta_{x_1} = \frac{\sigma_\eta}{\sqrt{2}} N_5$$

$$\eta_{x_2} = \frac{\sigma_\eta}{\sqrt{2}} N_6$$

where N_5, $N_6 \sim N(0,1)$ yields

$$W(t)\ V(t) = \frac{\sigma_S^2}{2}(N_1^2 + N_2^2)\cos\theta + \frac{\sigma_\eta^2}{2\sqrt{2}}(N_3N_5 + N_4N_6)$$

$$+ \frac{\sigma_S\sigma_\eta}{2\sqrt{2}}(N_1N_5 + N_2N_6) + \frac{\sigma_S\sigma_\eta}{2}(N_1N_3 + N_2N_4)\cos\theta\ .$$

Again, the quantity of interest is the average power relative to the noise power σ_η^2 so the equation is divided through by σ_η^2. Letting

$$\rho = \frac{\sigma_S^2}{\sigma_\eta^2}\ ,$$

$$OX = \frac{\overline{W(t)\ V(t)}}{\sigma_\eta^2} = \frac{\rho}{2}(N_1^2 + N_2^2)\cos\theta + \frac{1}{2\sqrt{2}}(N_3N_5 + N_4N_6)$$

$$+ \frac{\sqrt{\rho/2}}{2}(N_1N_5 + N_2N_6) + \sqrt{\frac{\rho}{2}}(N_1N_3 + N_2N_4)\cos\theta \ .$$

Once again, ρ must be weighted by γ_i, the fraction of time spent in frequency bin i. The expression for OX is now

$$OX_i = \frac{\gamma_i\rho}{2}(N_1^2 + N_2^2)\cos\theta + \frac{1}{2\sqrt{2}}(N_3N_5 + N_4N_6)$$

$$+ \frac{\gamma_i\rho/2}{2}(N_1N_5 + N_2N_6) + \frac{\sqrt{\gamma_i\rho}}{2}(N_1N_3 + N_2N_4)\cos\theta \ .$$

Similarly,

$$OY_i = \frac{\gamma_i\rho}{2}(N_1^2 + N_2^2)\sin\theta + \frac{1}{2\sqrt{2}}(N_3N_7 + N_4N_8)$$

$$+ \frac{\sqrt{\gamma_i\rho/2}}{2}(N_1N_7 + N_2N_8) + \frac{\sqrt{\gamma_i\rho}}{2}(N_1N_3 + N_2N_4)\sin\theta \ ,$$

where N_7, $N_8 \sim N(0,1)$.

3.3.3 Summary

The MAX-OR processor chooses the frequency bin on the basis of $\max\{OO_j\}$ where j ranges over the number of bins. The bearing estimate is computed using the values of OX and OY for that bin. The output of the processor depicted in Figure 3.3.2.1 may be modeled with the proper statistics by generating

eight samples of a standard normal distribution and using the above equations. The algorithm model of this processor is depicted in Figure 3.3.3.

3.3.4 Signal-to-Noise Ratios and Variances

The data used in the simulation was generated by an algorithm designed to generate data with statistical properties closely resembling those of the output of a general processor for DIFAR buoys. In order to generate data by this scheme, knowledge of the signal-to-noise ratio at the input to the comb filter bank is required. The signal-to-noise ratio was computed at each time point from

$$[S/N]_i = S - N - 20 \log_{10} R_i$$

where

$[S/N]_i$ is the signal-to-noise ratio in dB at time t_i

S is the source level in dB

N is the noise level in dB

R_i is the range in yards at t_i.

The signal co-noise ratio is input into the data generation algorithm. This ratio is used in the scheme to generate the average power in each filter bin over the integration time. At the end of the data generation algorithm, when a frequency bin has been chosen to estimate the signal

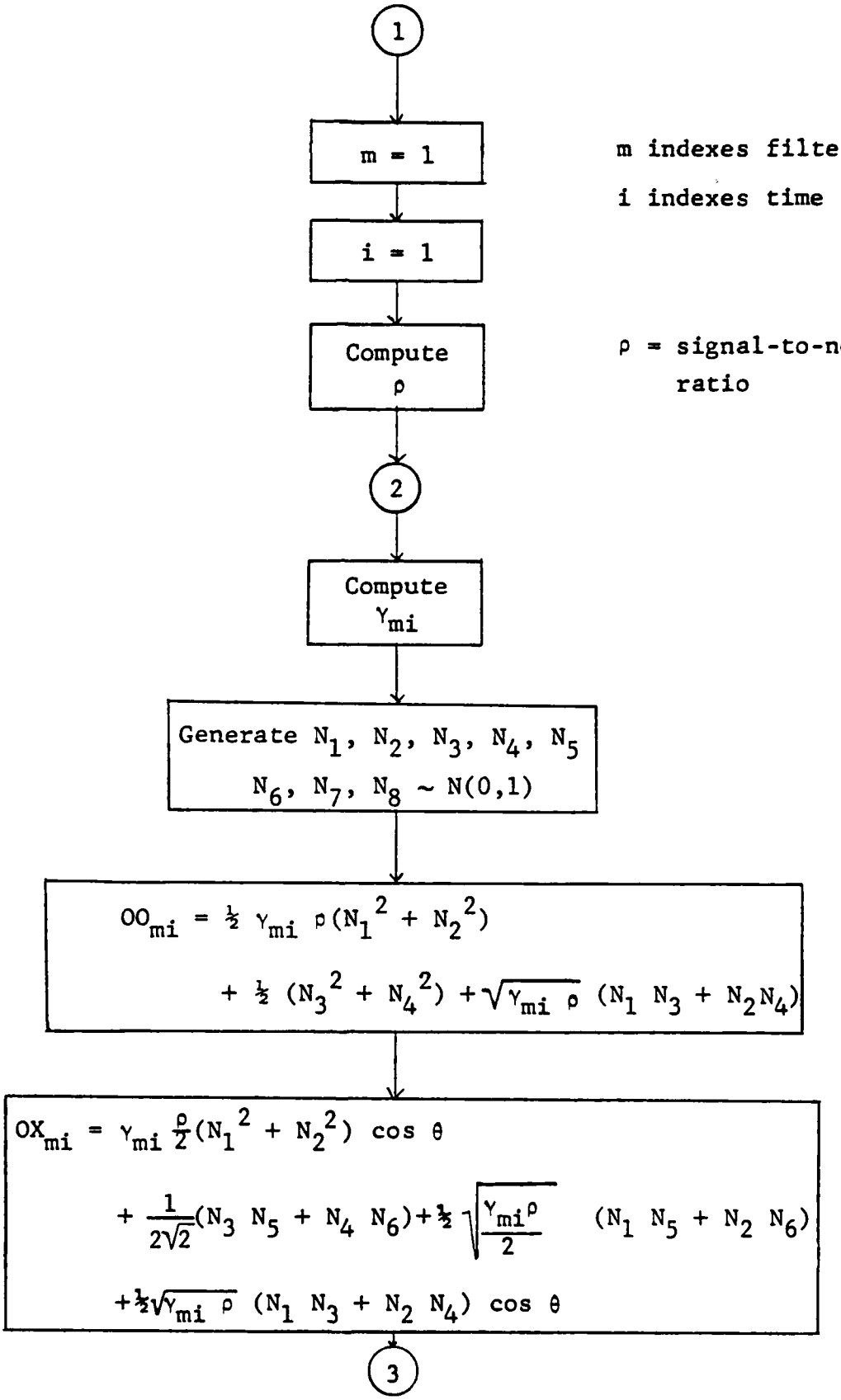


FIGURE 3.3.3 - FLOW CHART OF PROCESSOR

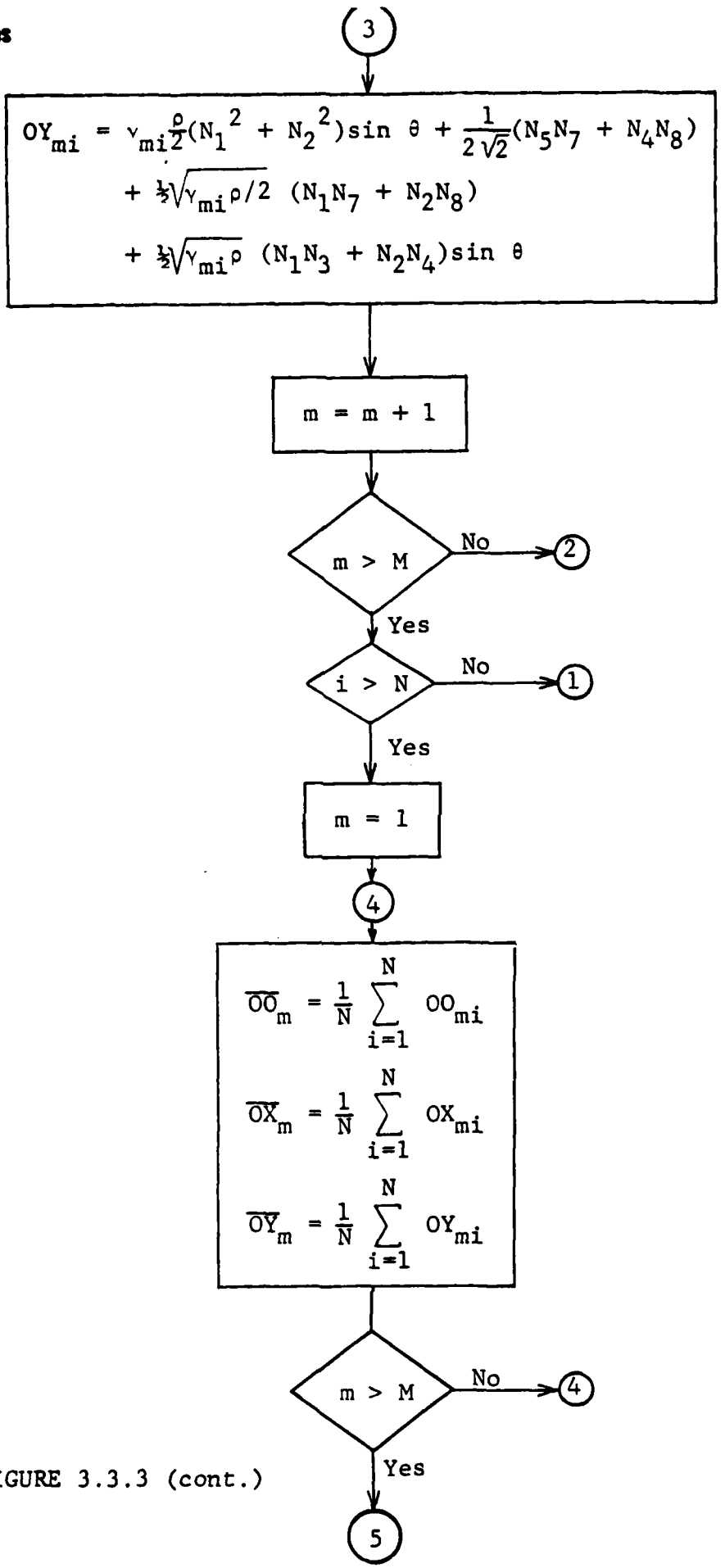


FIGURE 3.3.3 (cont.)

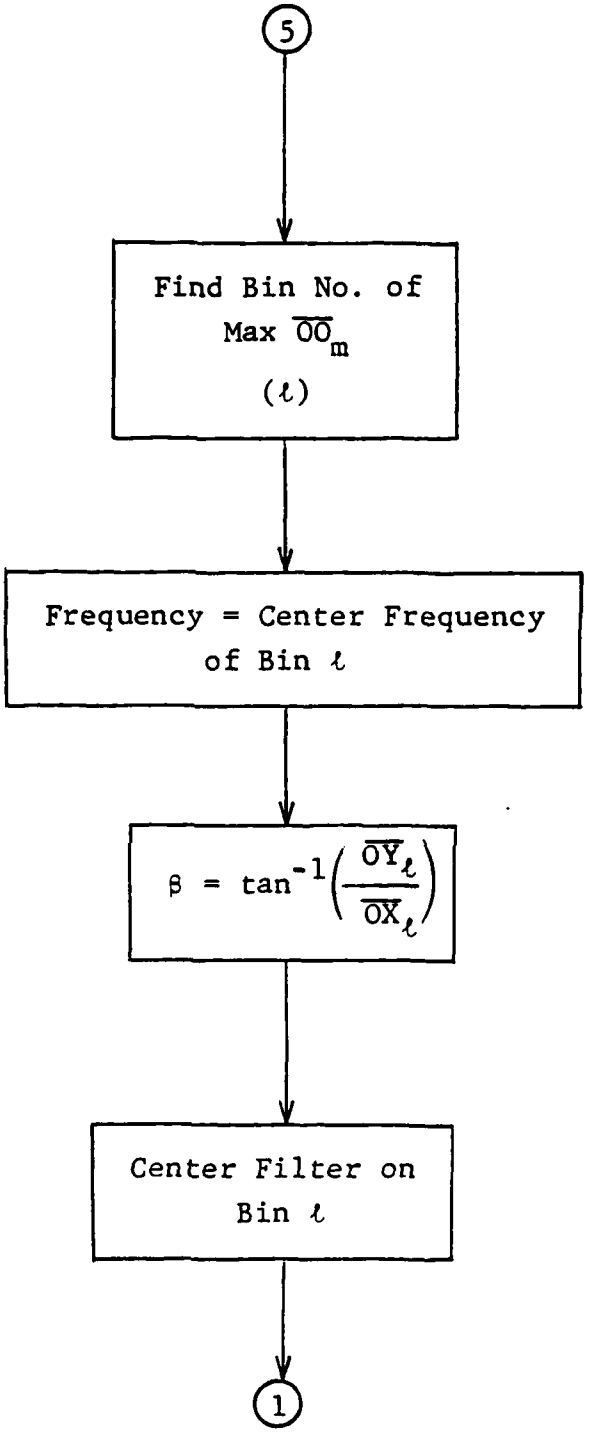


FIGURE 3.3.3 (Continued)

frequency, an estimate of the signal-to-noise ratio is made using only the average power in the selected bin of the comb filter and the knowledge of the average noise power.

$$[S/N]_i = 10 \log_{10}\left(\frac{(S+N) - (\bar{N})}{(\bar{N})}\right)$$

Letting OO_{max_i} represent S+N in the selected bin and remembering that all power values are relative to the noise power, the expression becomes

$$S/N_i = 10 \log_{10}\left(\frac{OO_{max_i} - 1.}{1.}\right)$$

where OO_{max_i} is measured in power units (not dB). This method of estimating the signal-to-noise ratio was chosen to make the simulation as realistic as possible.

The MLP, hybrid, and iterated sequential algorithms require that the covariance matrix of the data be supplied. Since the data points are independent, the covariance matrix is diagonal and all that needs to be found is an estimation of the variance of the population from which each data point is obtained. It is clear that the population and the associated variances are functions of the signal-to-noise ratio. Estimates of the variances as a function of signal-to-noise ratio were found by generating 1,000 samples for $\Delta f = .1$ Hz at constant [S/N] for several different [S/N] ratios. Since the value of the frequency is only resolvable to one bin width, the frequency variance was given a lower bound of $\frac{\Delta f^2}{12}$, the variance of a variate with density function constant over the interval

Δf and zero everywhere else. These numbers are correct when Δf = .1 Hz. For other values of Δf the transformation

$$\sigma_{\Delta f} = (10)\ (\Delta f)\ (\sigma_{10})$$

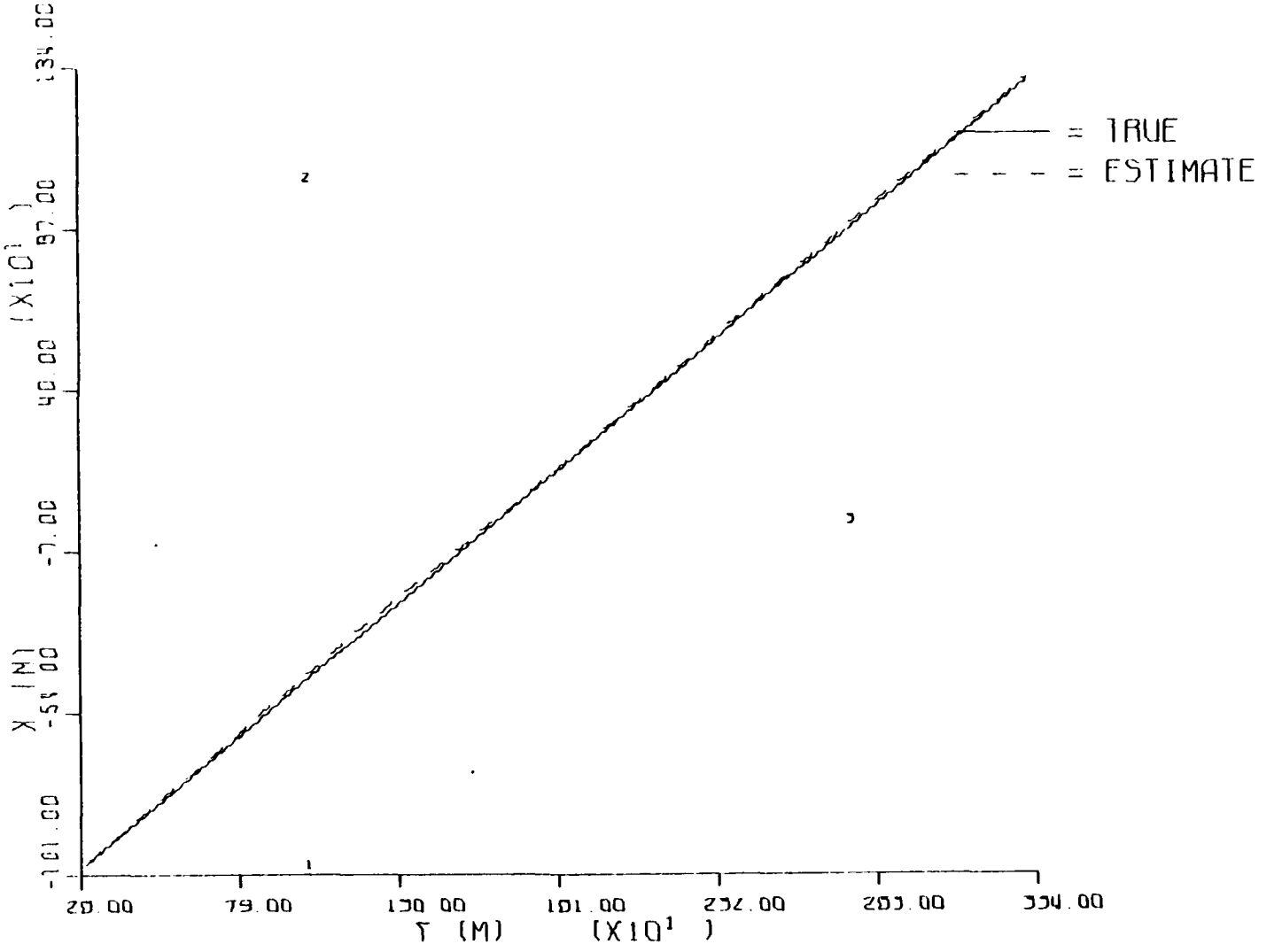
is used where σ_{10} is the standard deviation of the frequency data for a particular signal-to-noise ratio when Δf = .1 Hz. The bearing variances do not depend on Δf.

3.4 Test Results and Analysis

3.4.1 Introduction

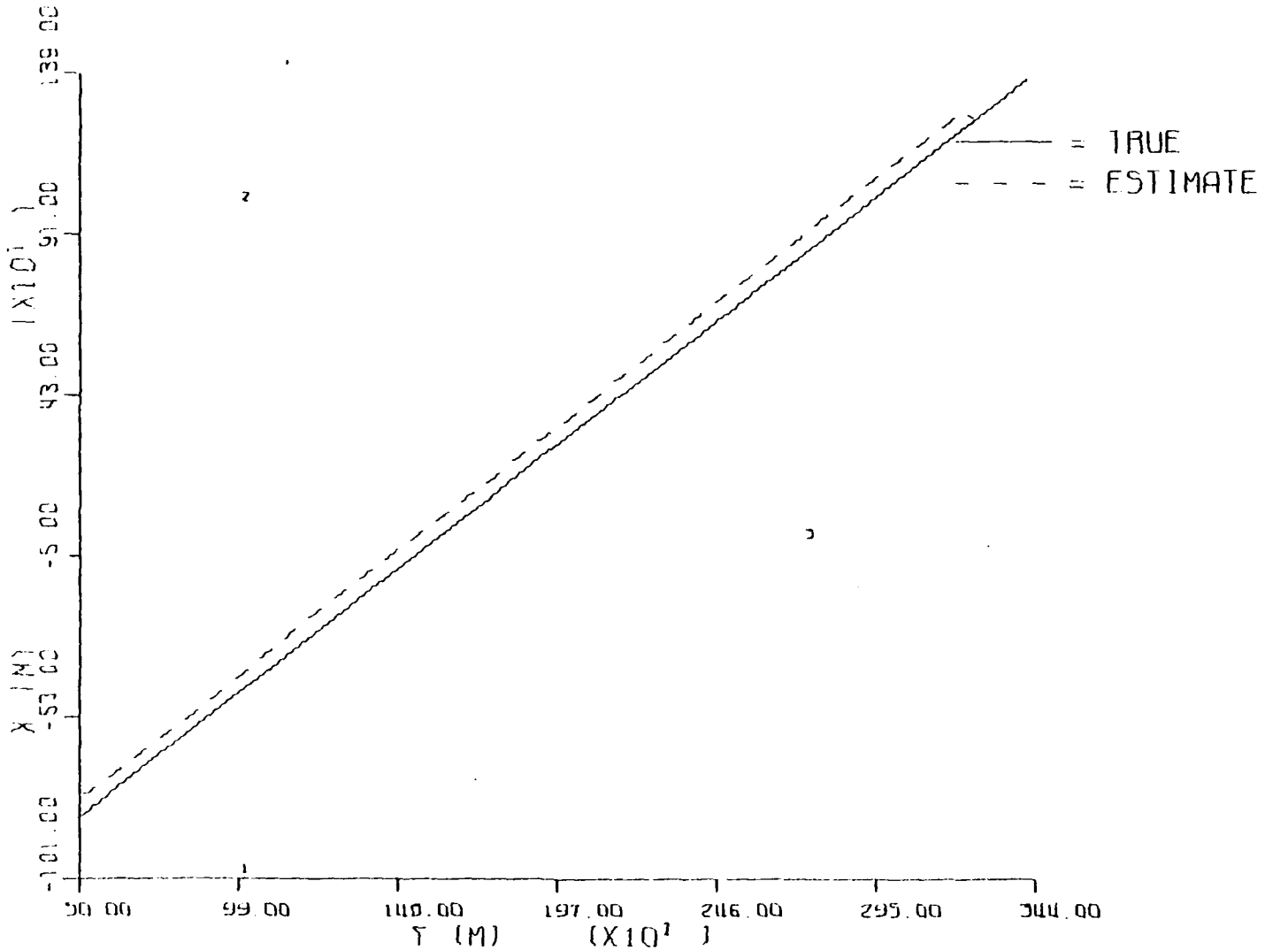
This section presents the results of applying the hybrid, MLP, and sequential tracking algorithms to the four scenarios described in Section 3-2. To help understand the effectiveness of the algorithms, several plots and charts have been prepared for each scenario. Figures 3.4.1 through 3.4.4 contain the average estimated trajectory for each scenario, which are found by averaging over all Monte Carlo runs the position vectors for each time point that a state estimate is generated.

Figures 3.4.5 through 3.4.8 present graphs of the average distance error, over all Monte Carlo runs, for a given scenario. It is computed by finding the difference between the estimated and true trajectories for each Monte Carlo run and then averaging them. The dotted lines represent the one-sigma deviation limits about each average. Note, the tighter these bands are around the average, the smaller the trajectory error variation about the average trajectory error, indicating that the algorithm has converged to the true trajectory.



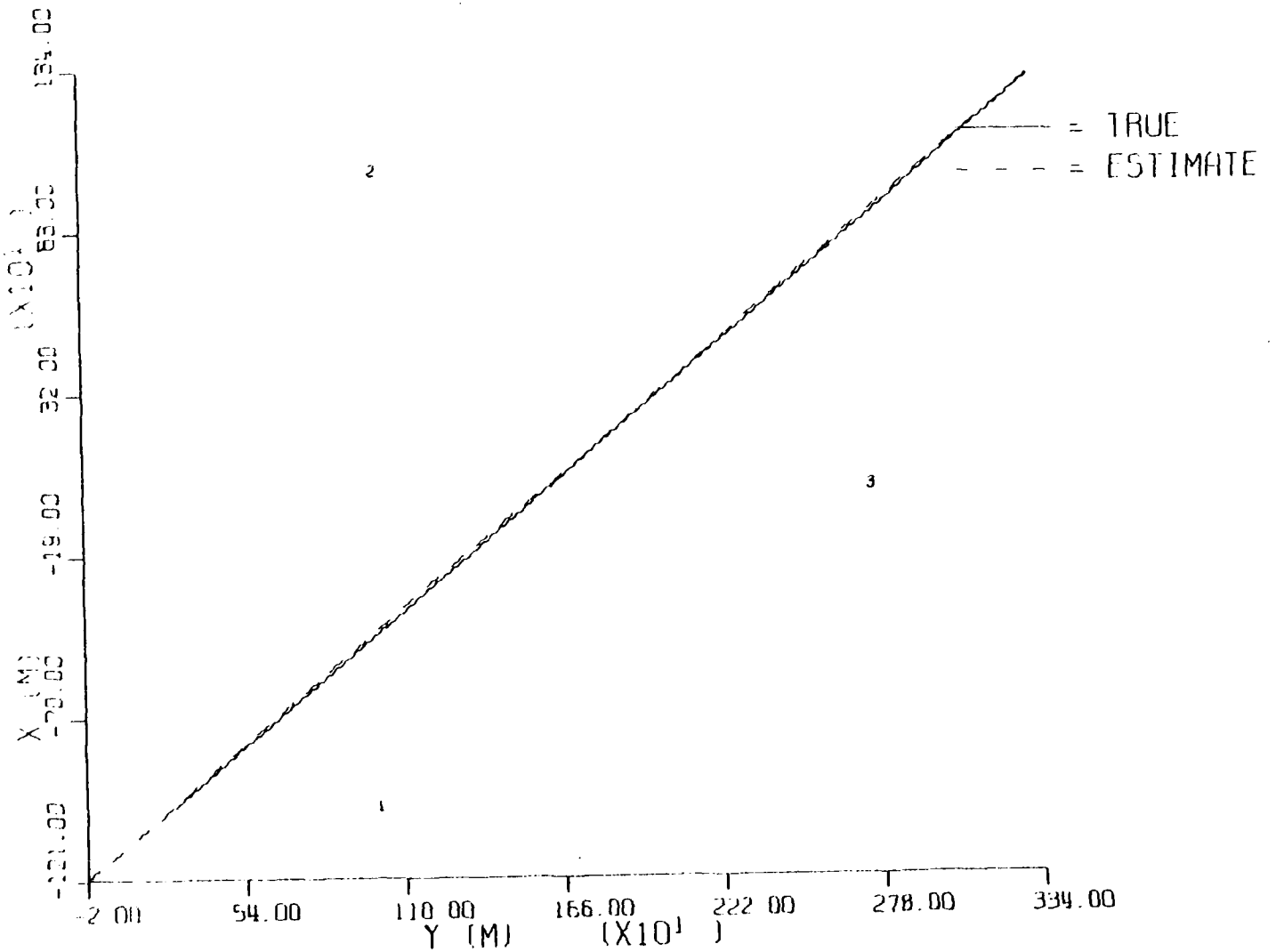
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.1a - HYBRID - SCENARIO 1



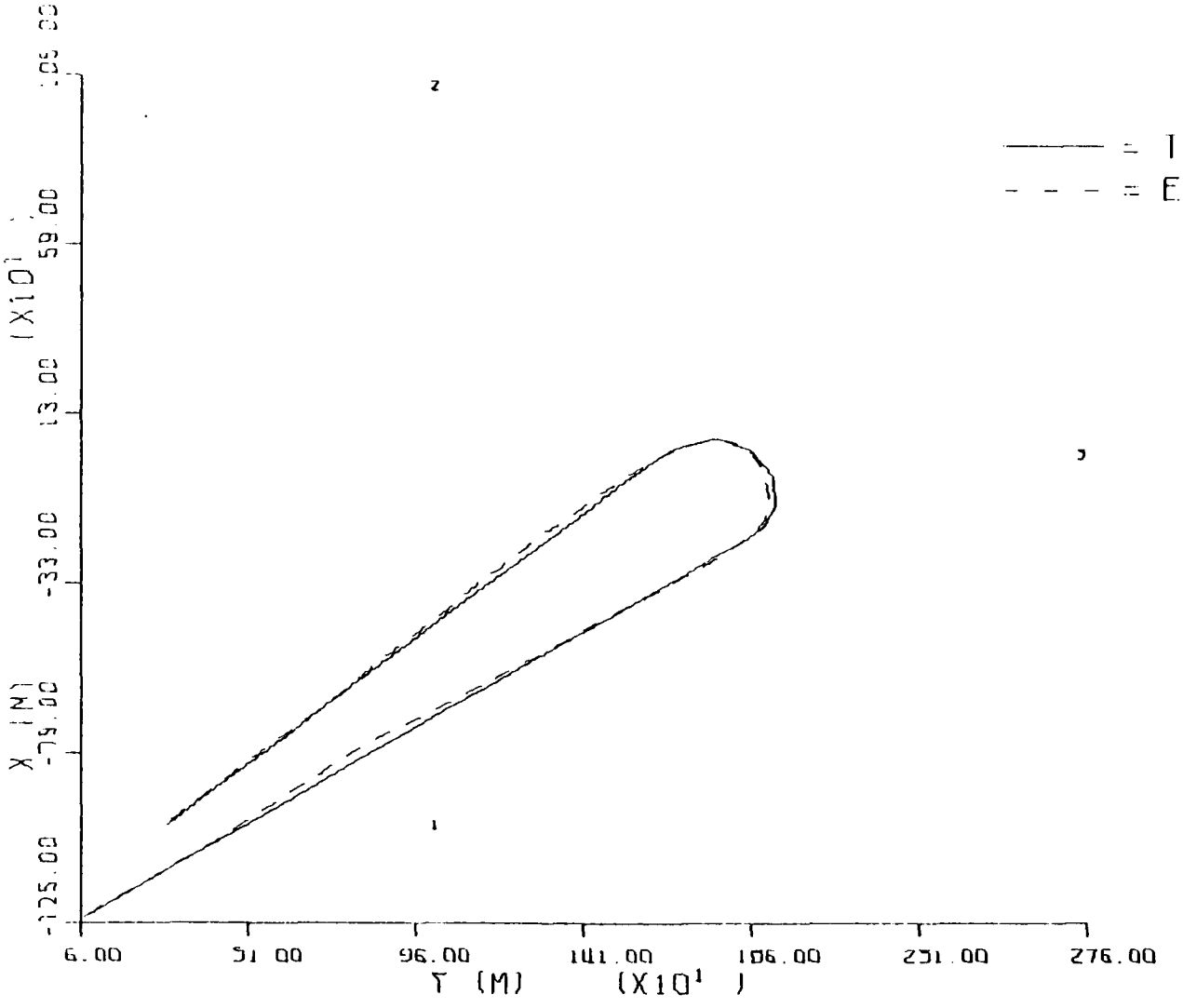
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.1b - MLP - SCENARIO 1



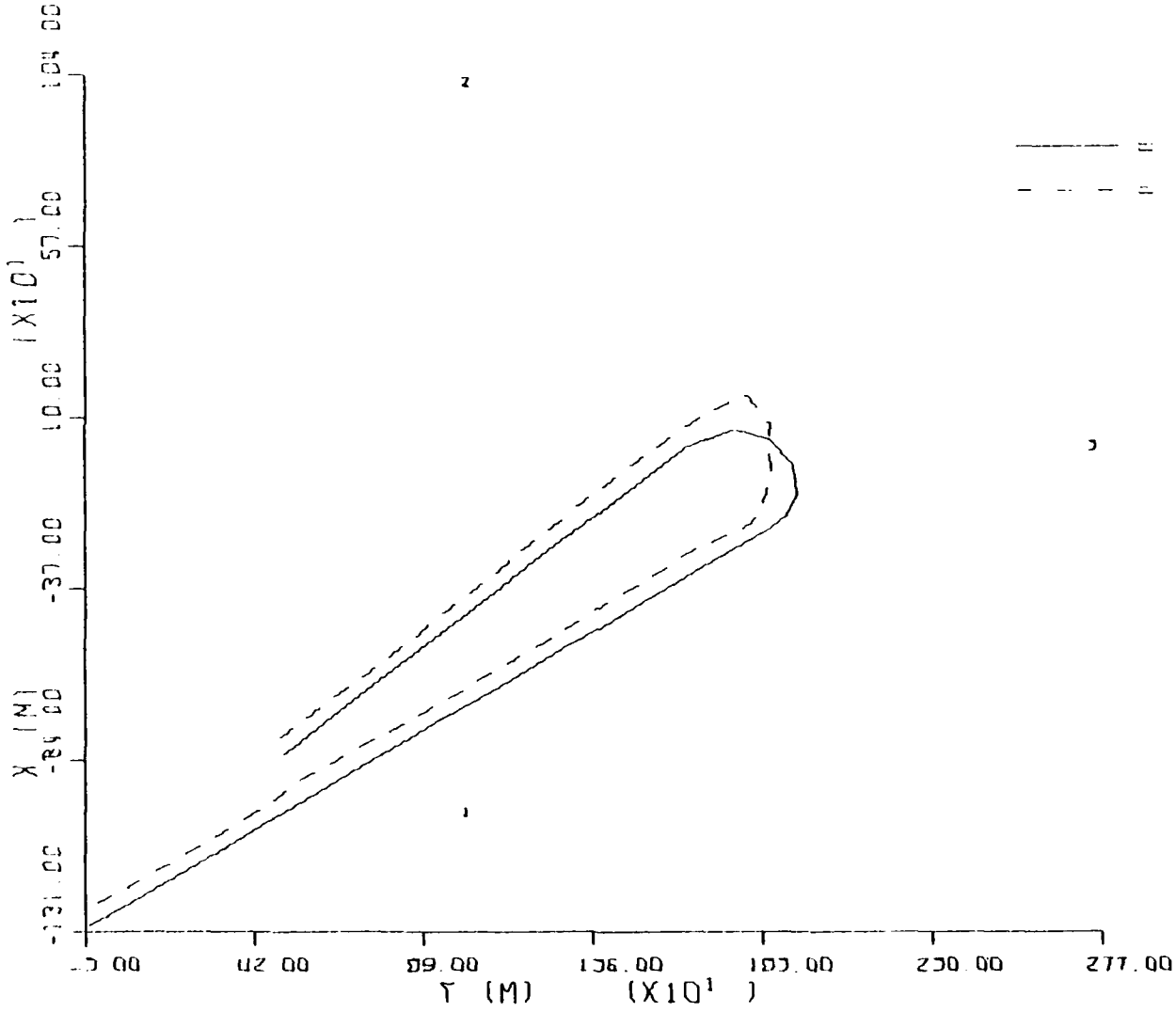
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.1c - SEQUENTIAL - SCENARIO 1



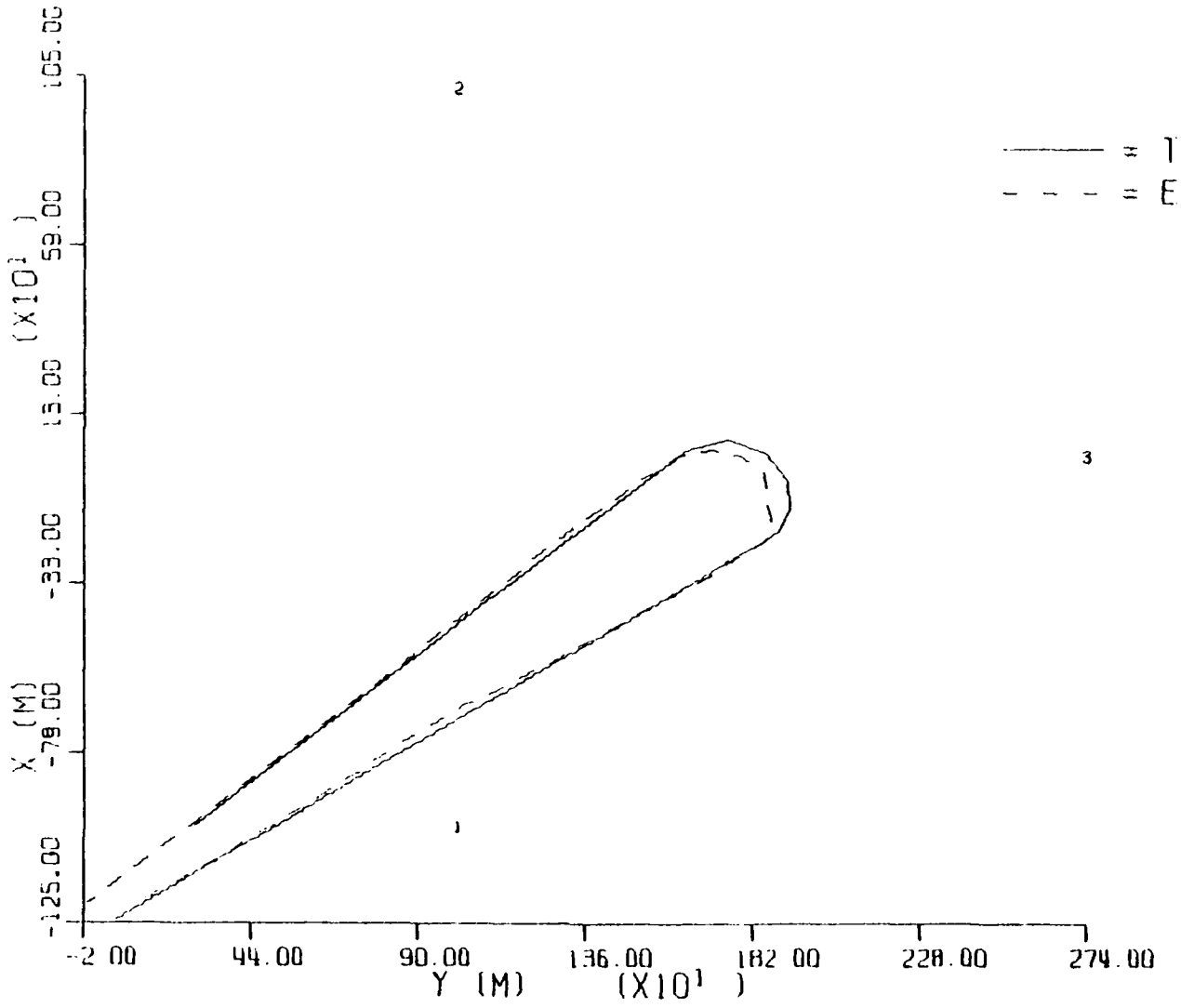
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.2a - HYBRID - SCENARIO 2



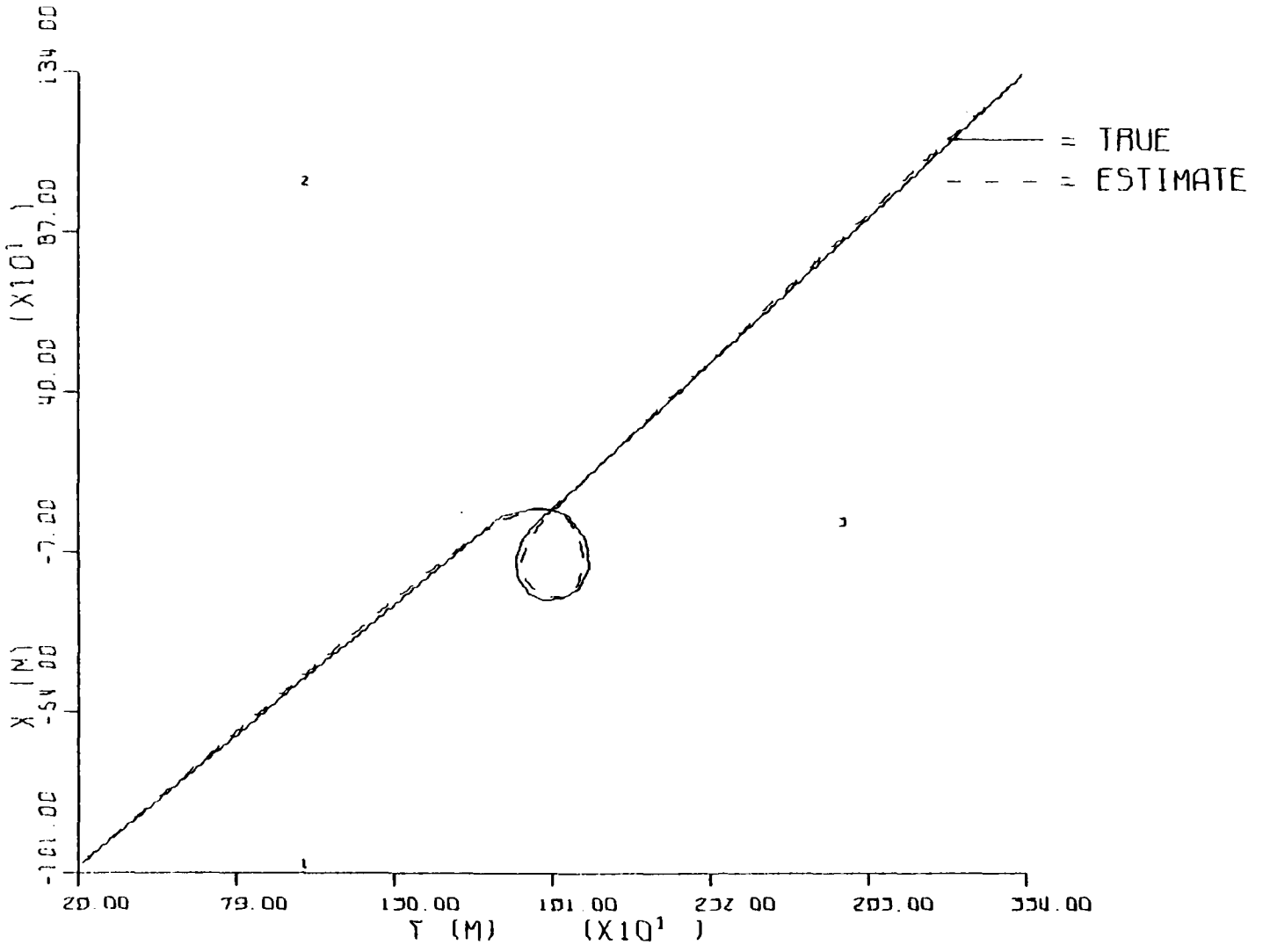
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.2b - MLP - SCENARIO 2



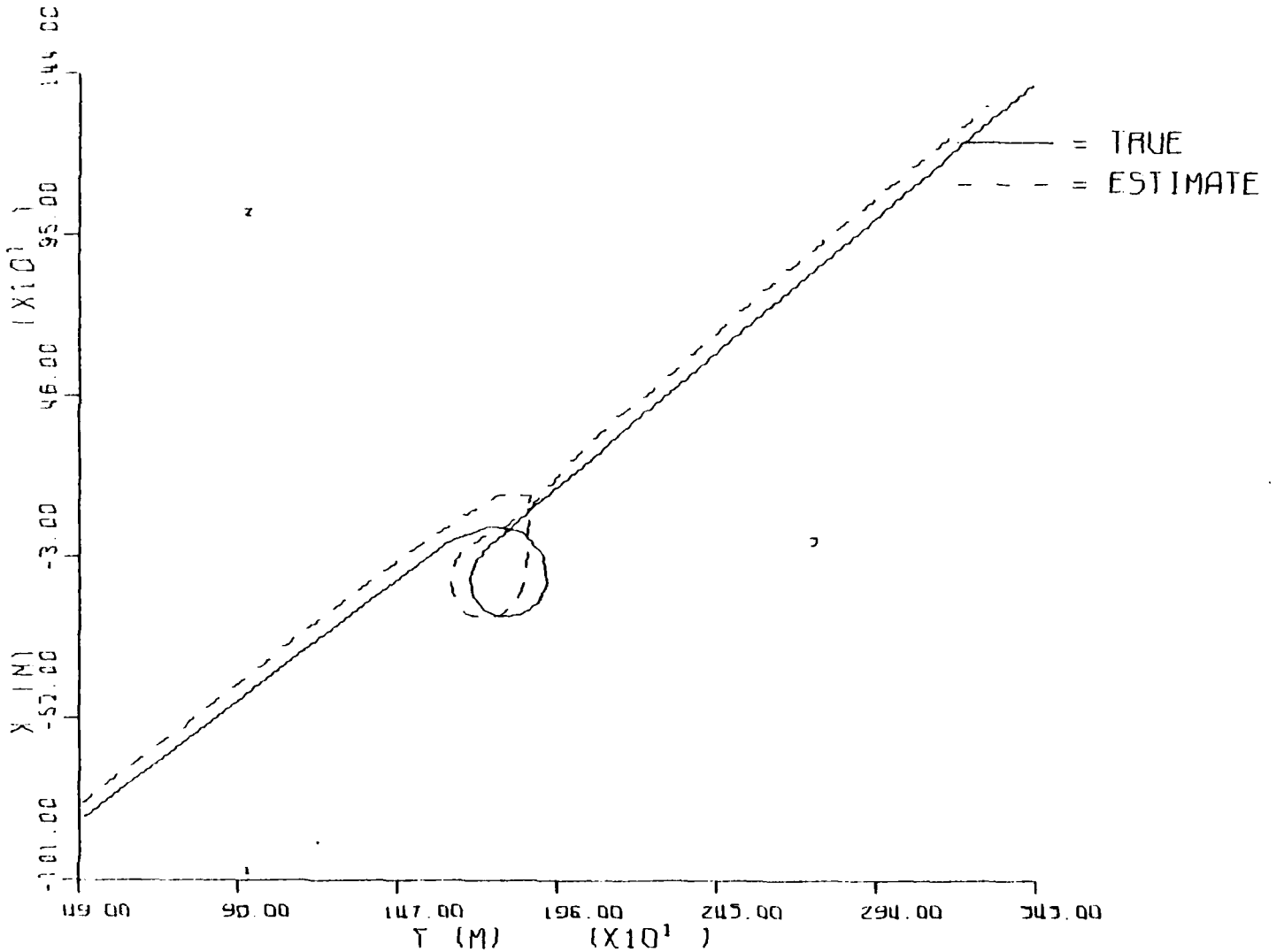
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.2c - SEQUENTIAL - SCENARIO 2



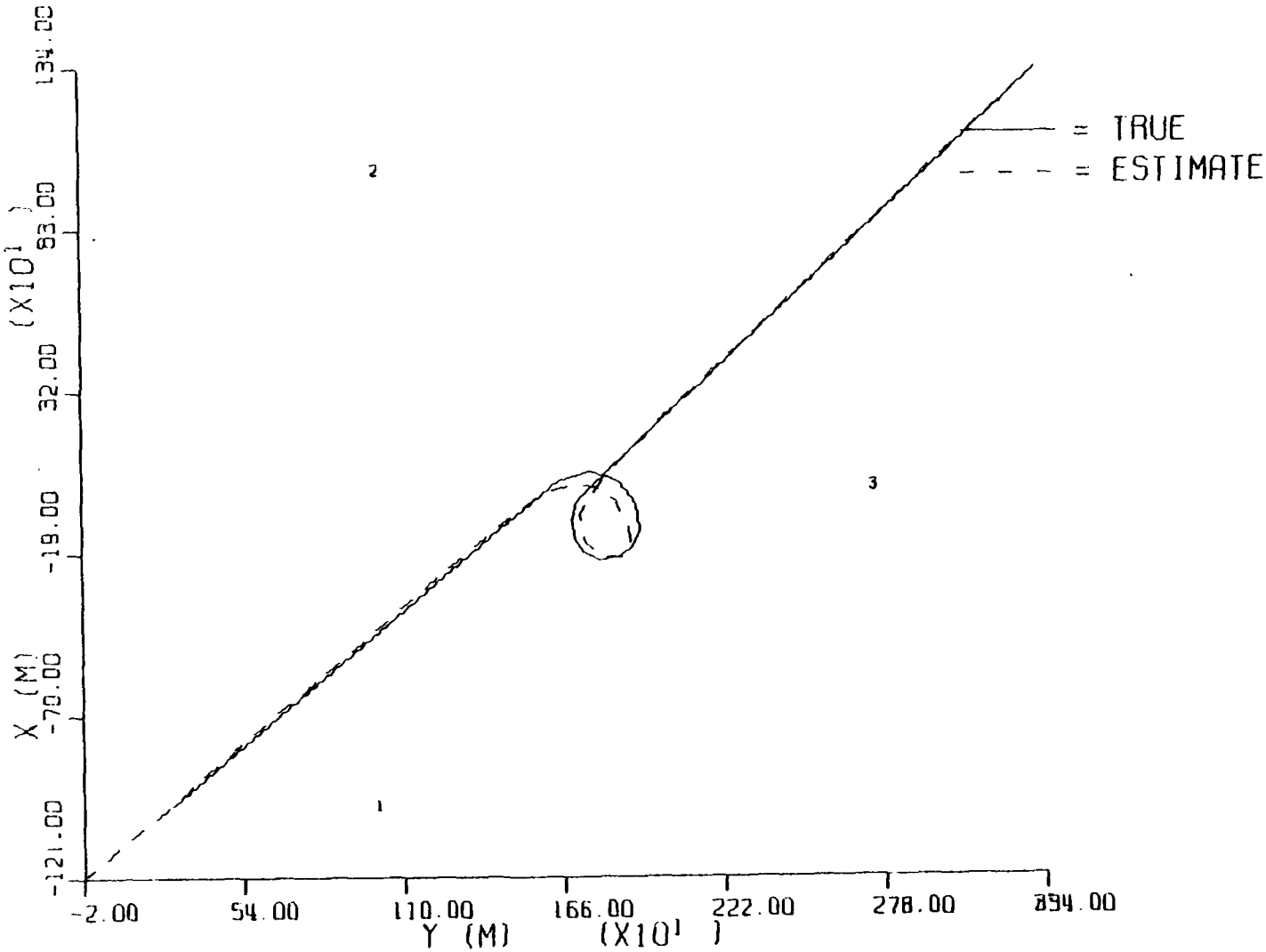
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.3a - HYBRID - SCENARIO 3



TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.3b - MLP - SCENARIO 3



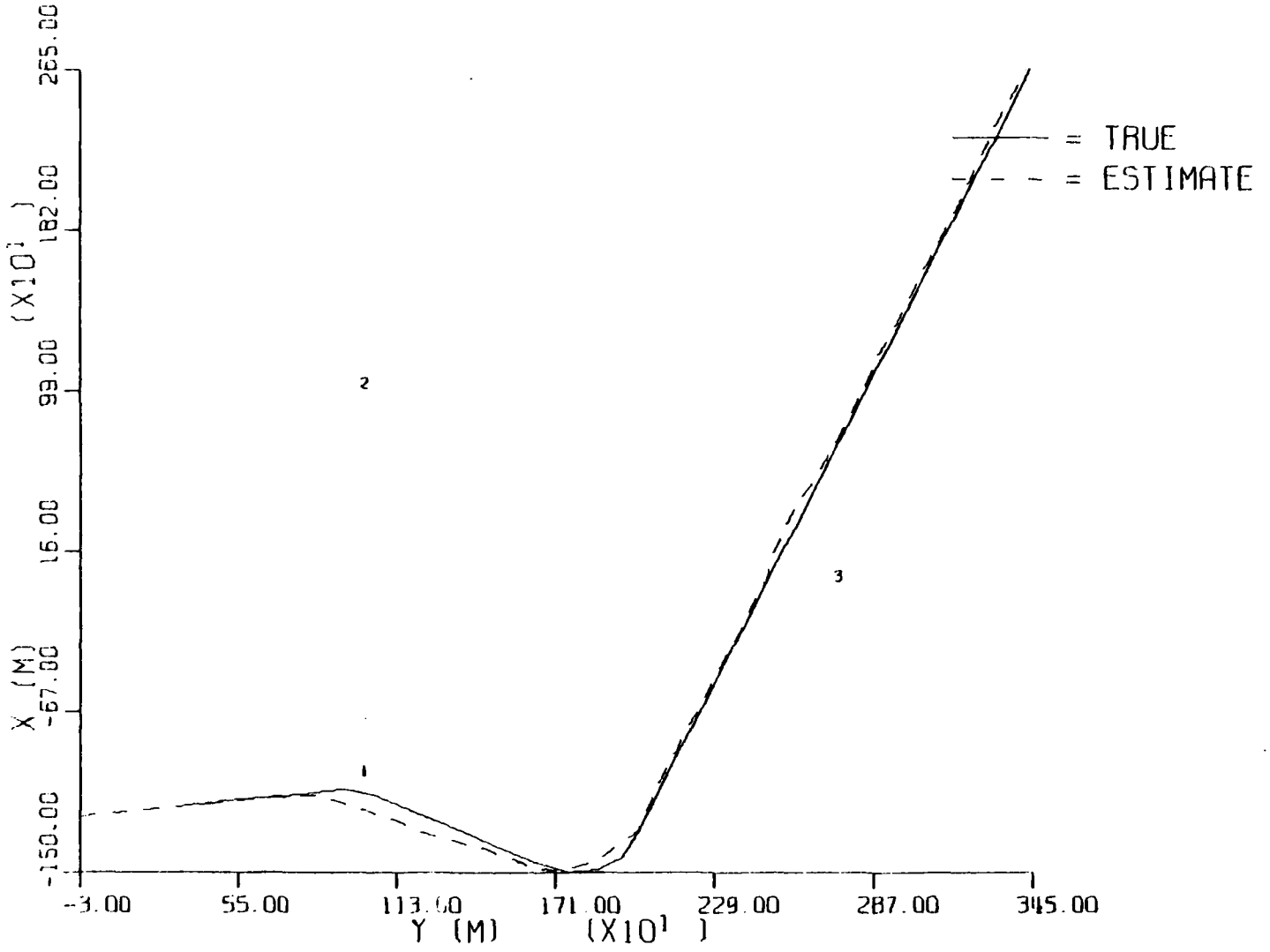
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.3c - SEQUENTIAL - SCENARIO 3

TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

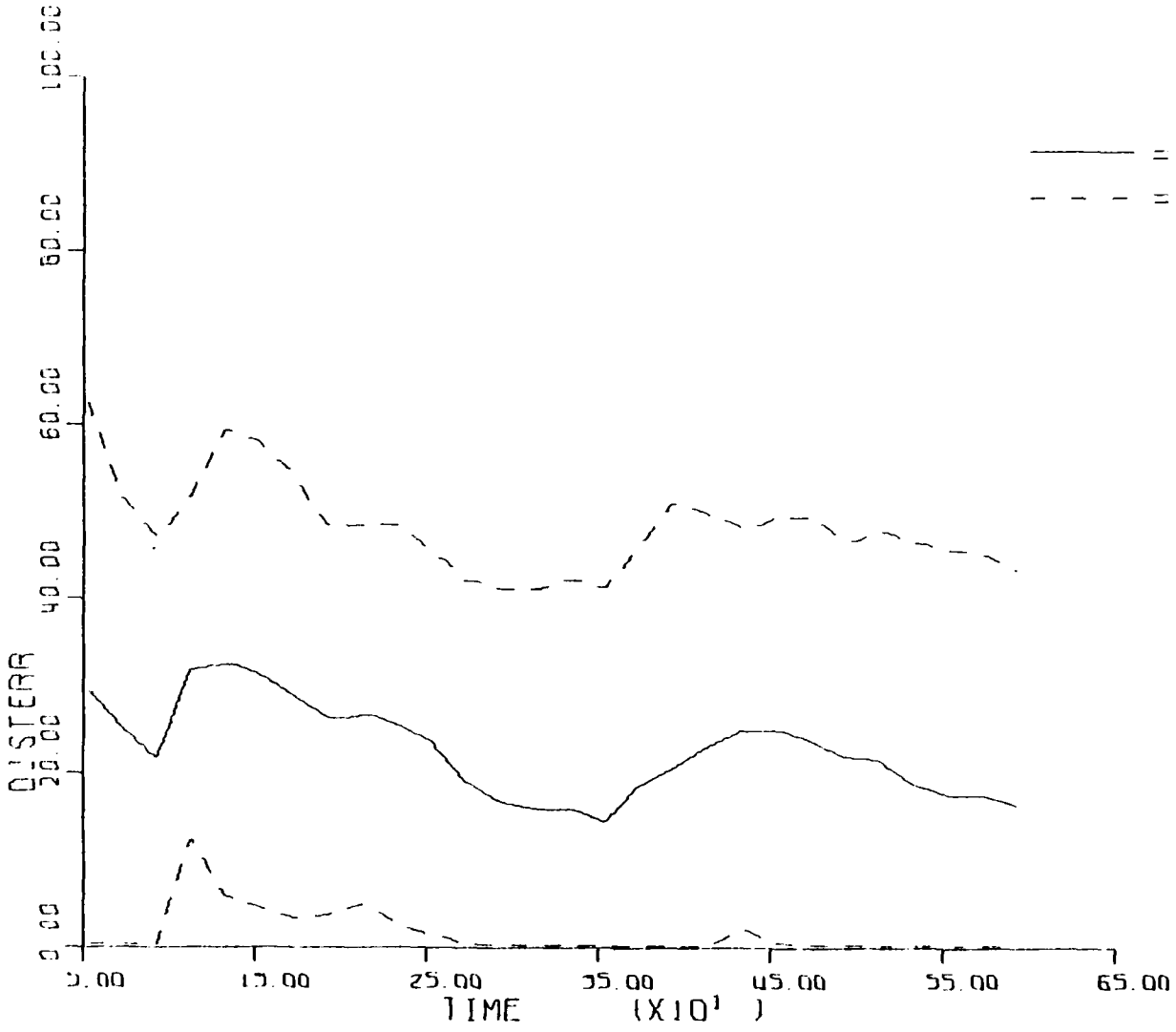
FIGURE 3.4.4a - HYBRID - SCENARIO 4

FIGURE 3.4.4b - MLP - SCENARIO 4



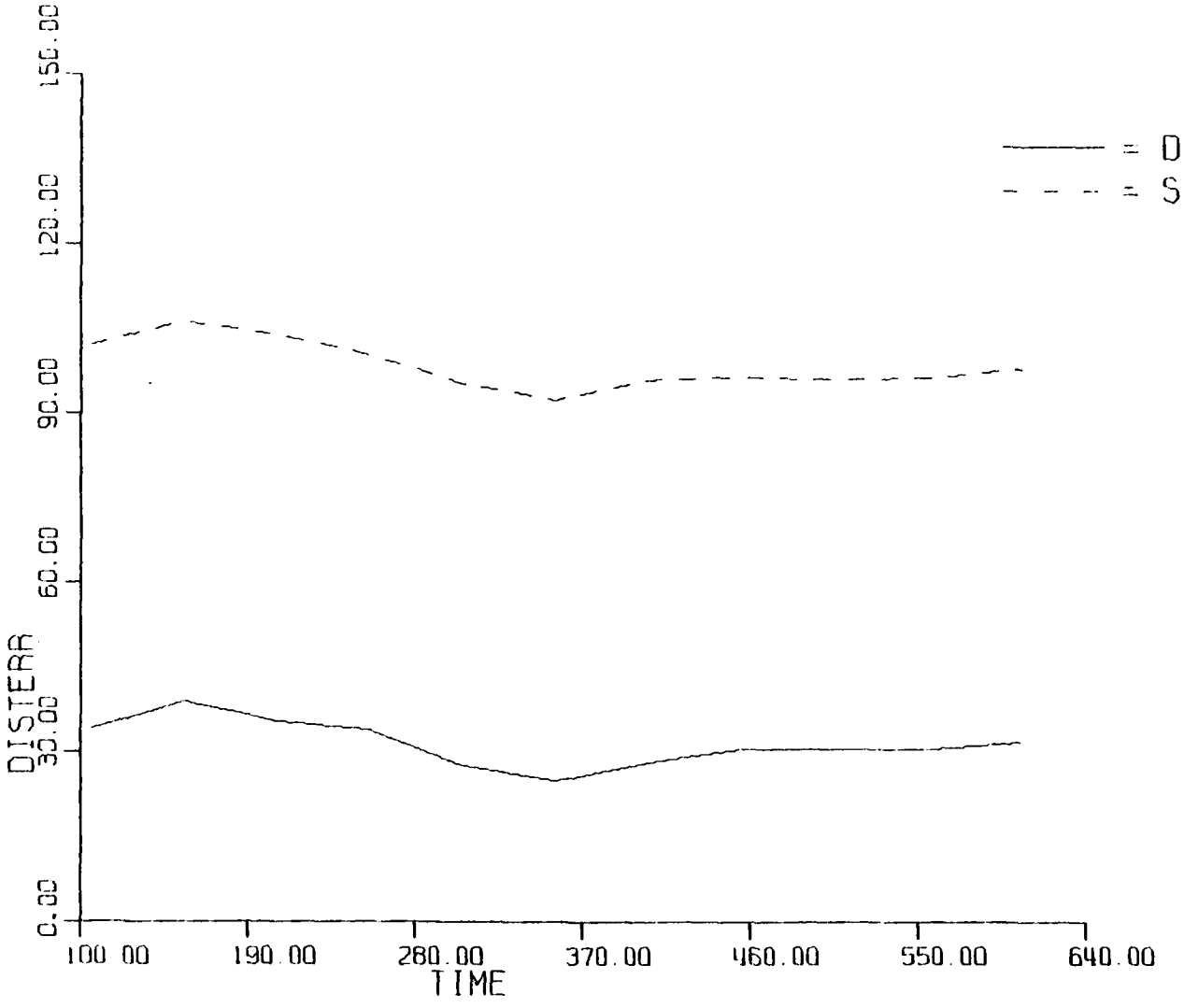
TRUE AND ESTIMATED TRAJECTORY WITH SENSORS

FIGURE 3.4.4c - SEQUENTIAL - SCENARIO 4



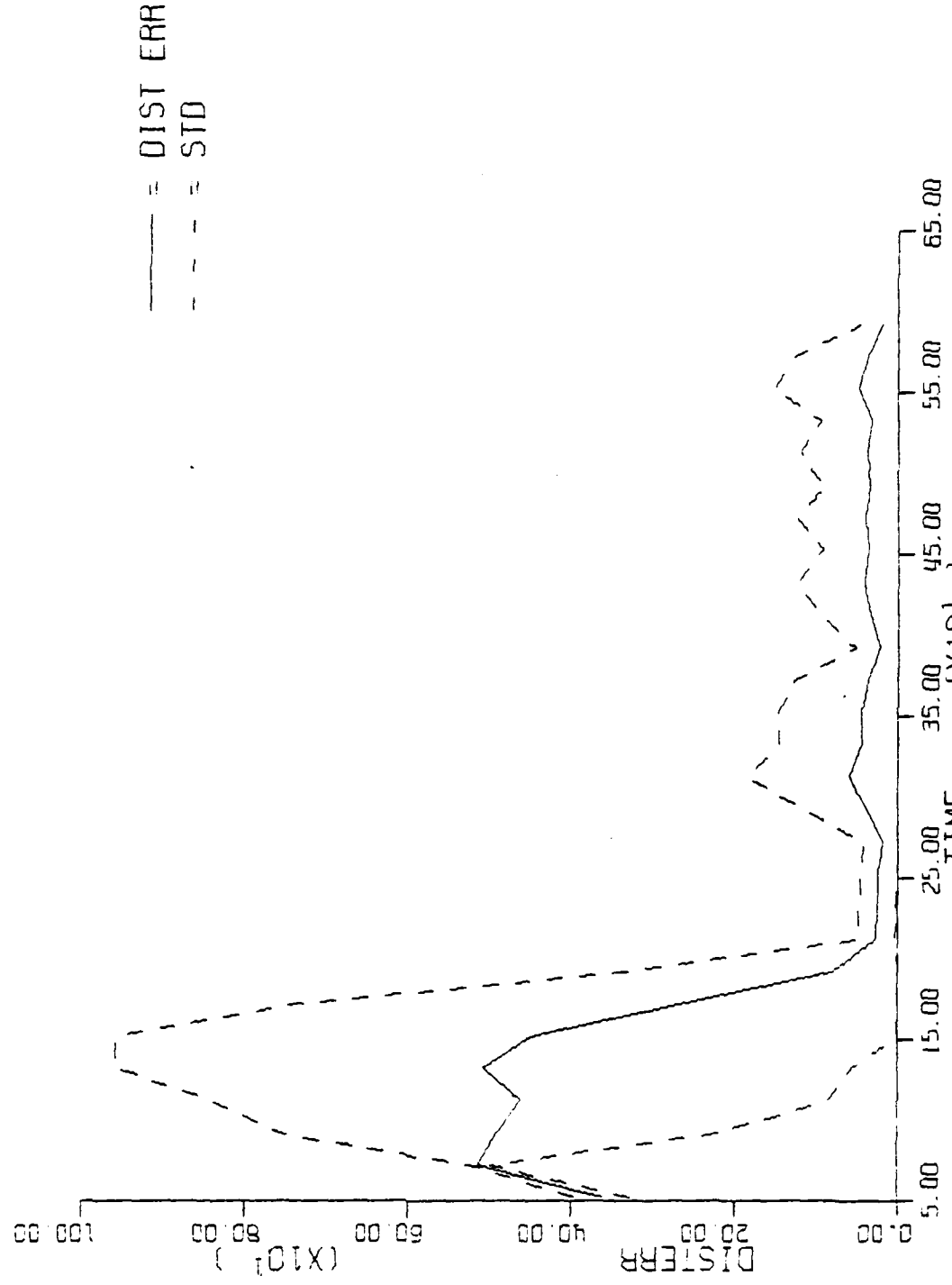
DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.5a - HYBRID - SCENARIO 1



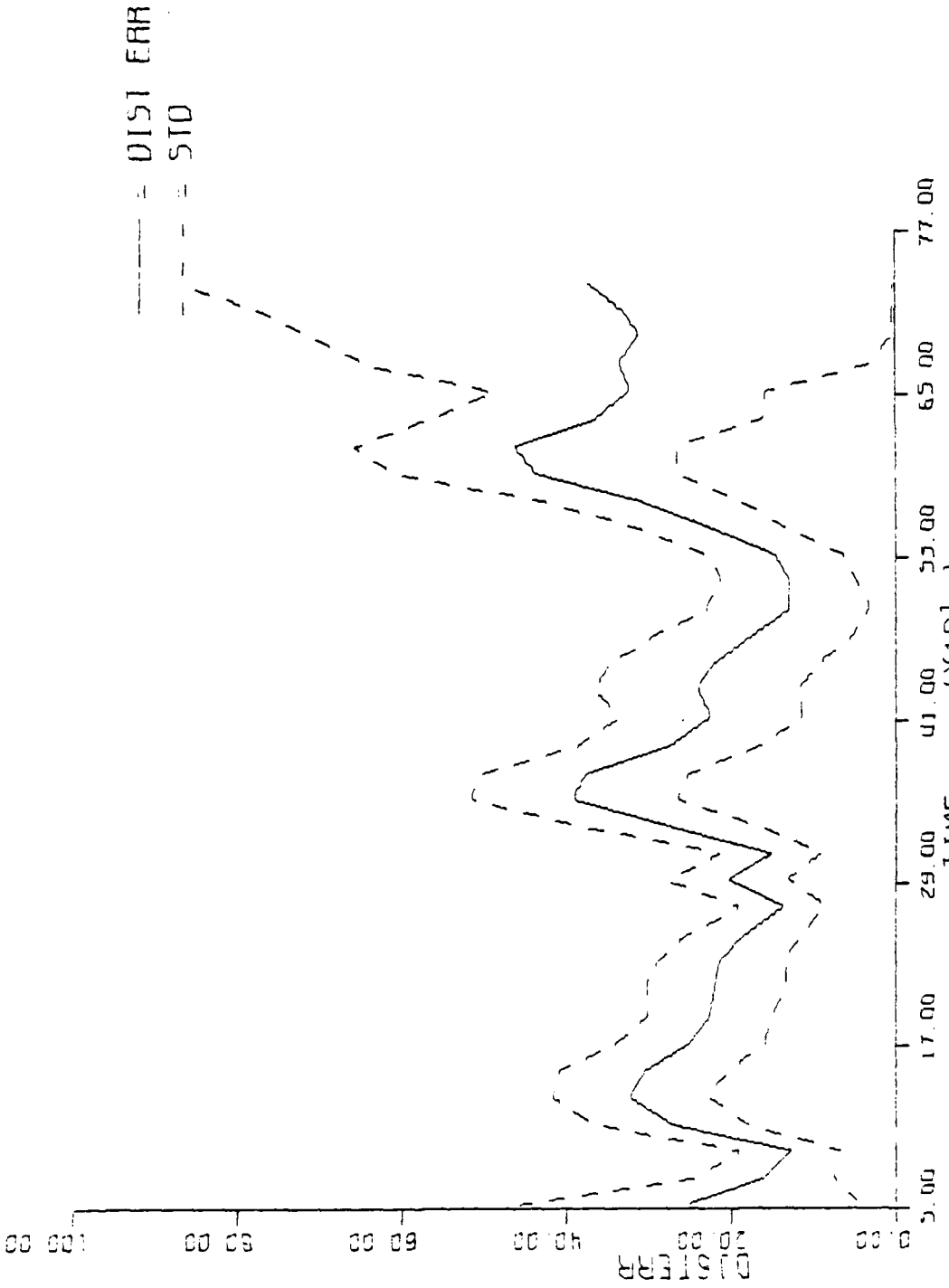
DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.5b - MLP - SCENARIO 1



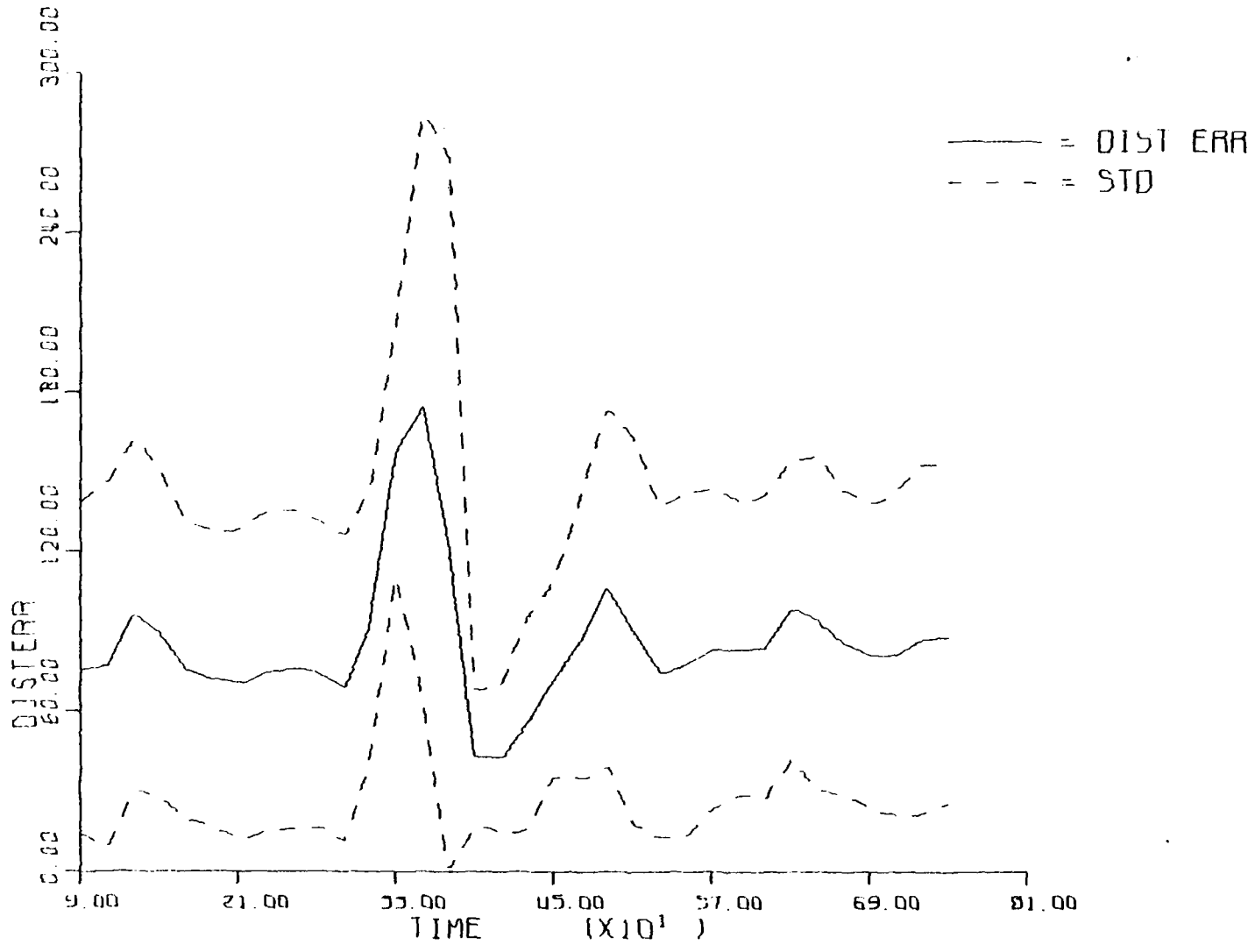
DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.5c - SEQUENTIAL - SCENARIO 1



DISTANCE ERROR AND STANDARD DEVIATION

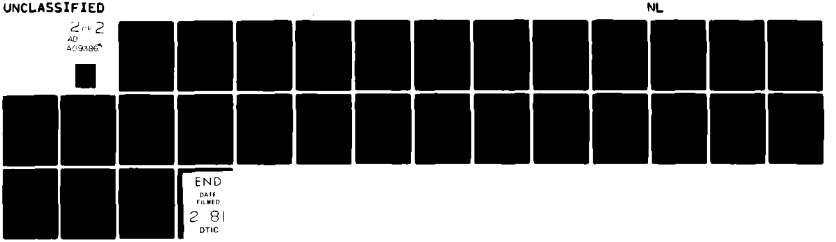
FIGURE 3.4.6a - HYBRID - SCENARIO 2



DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.6b - MLP - SCENARIO 2

AD-A093 867 TRACOR APPLIED SCIENCES AUSTIN TX F/G 17/1
HYBRID PASSIVE TRACKING ALGORITHMS.(U)
OCT 80 G CORSER, T WILSON N00014-78-C-0670
UNCLASSIFIED NL



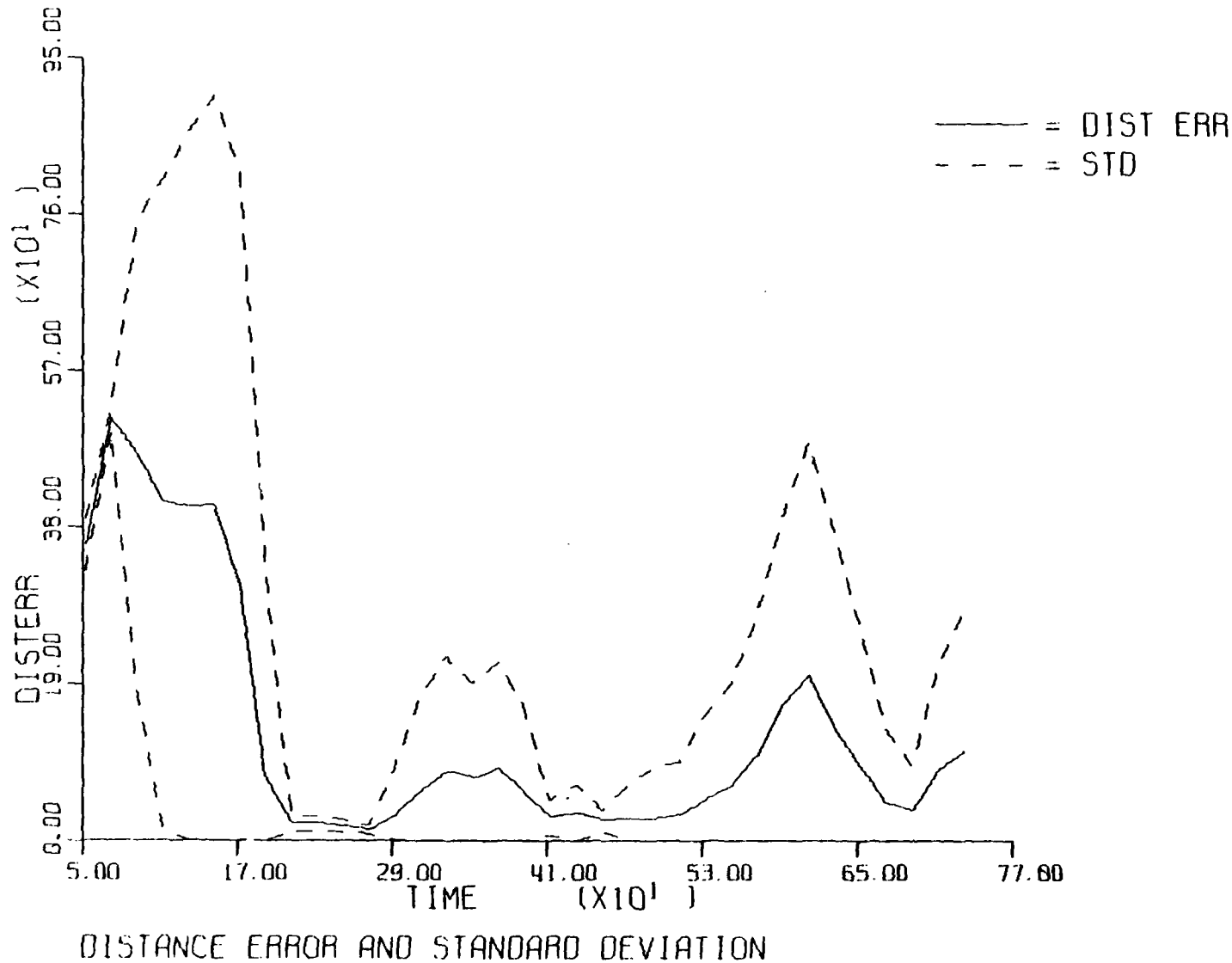
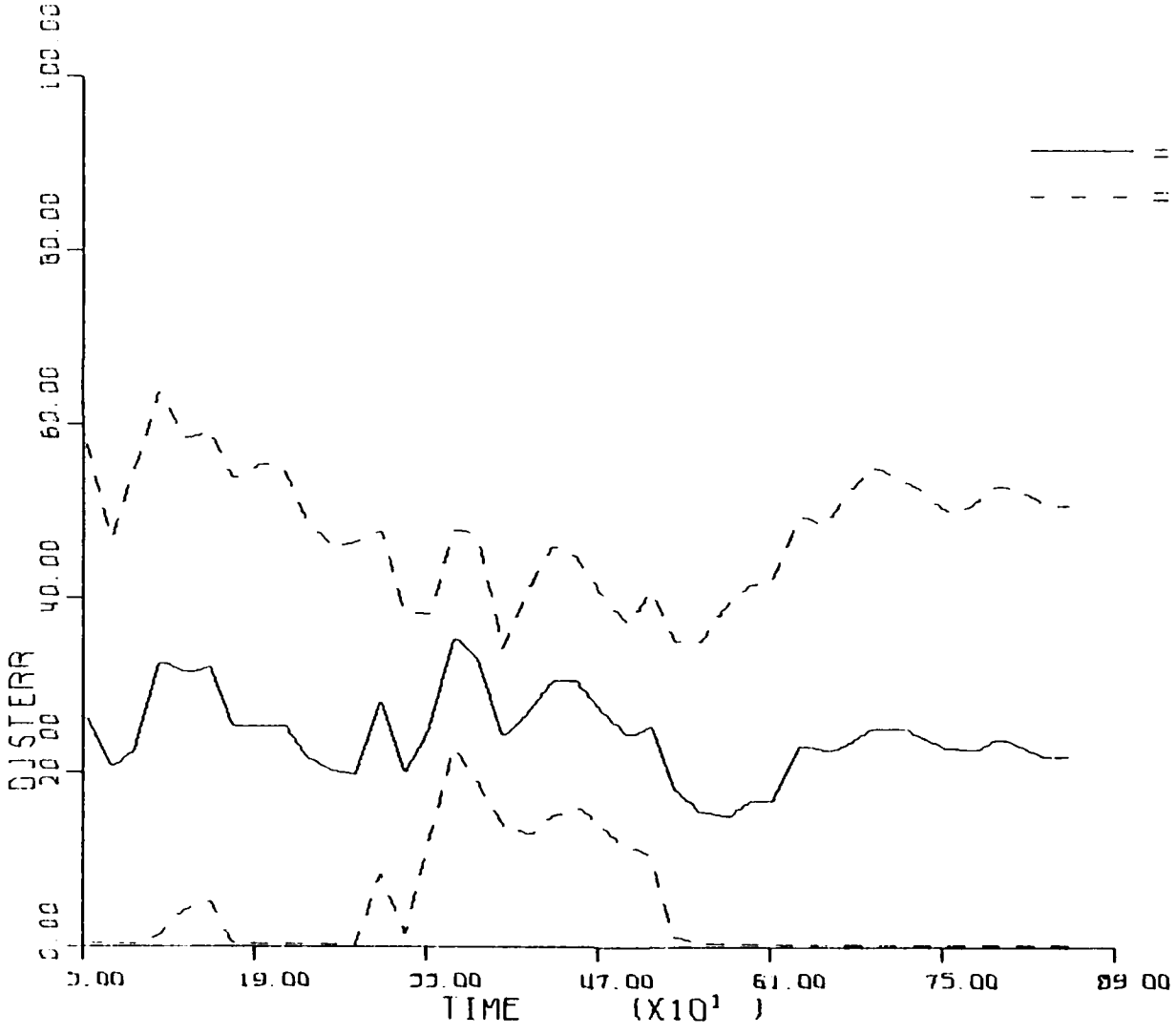
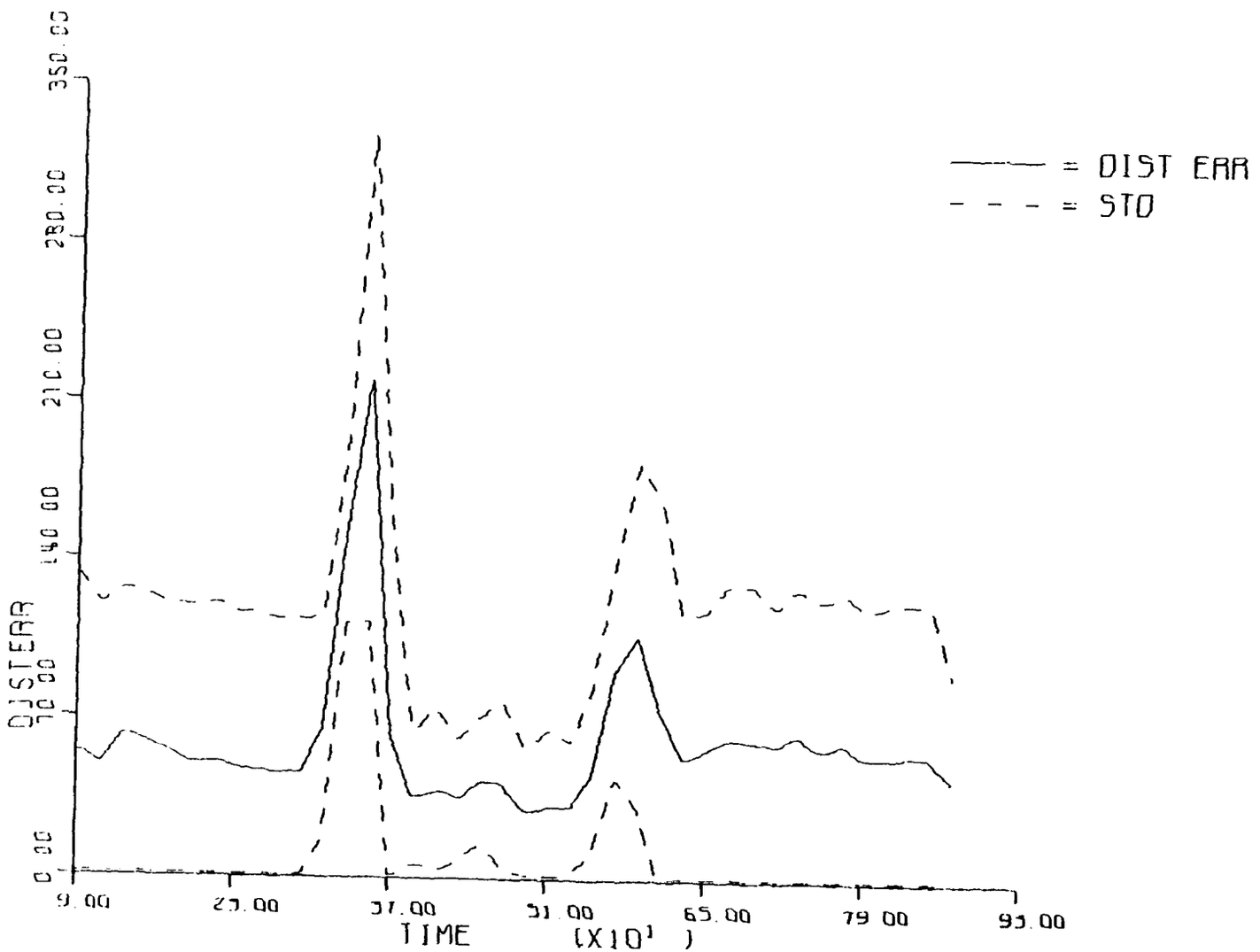


FIGURE 3.4.6c - SEQUENTIAL - SCENARIO 2



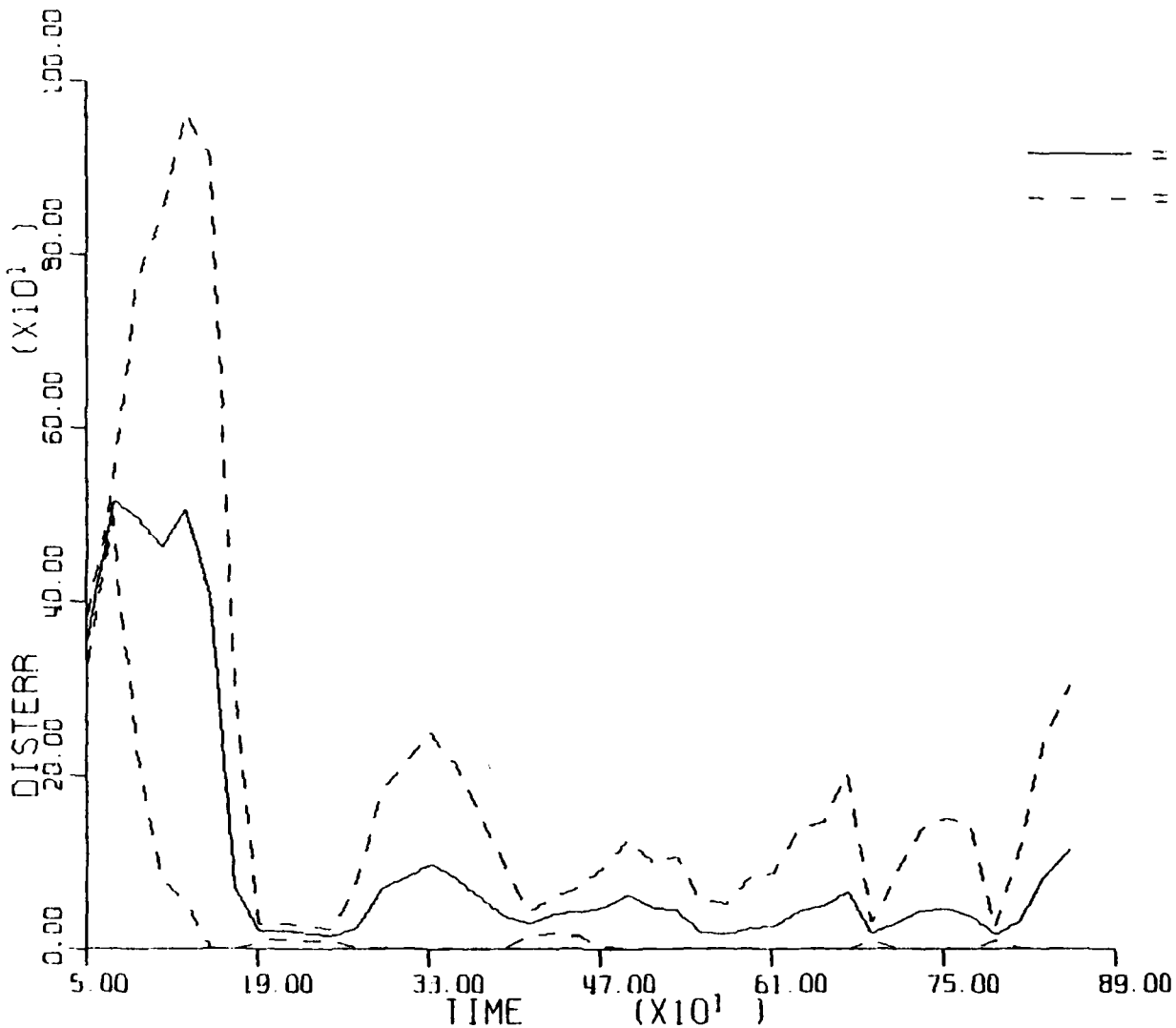
DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.7a - HYBRID - SCENARIO 3



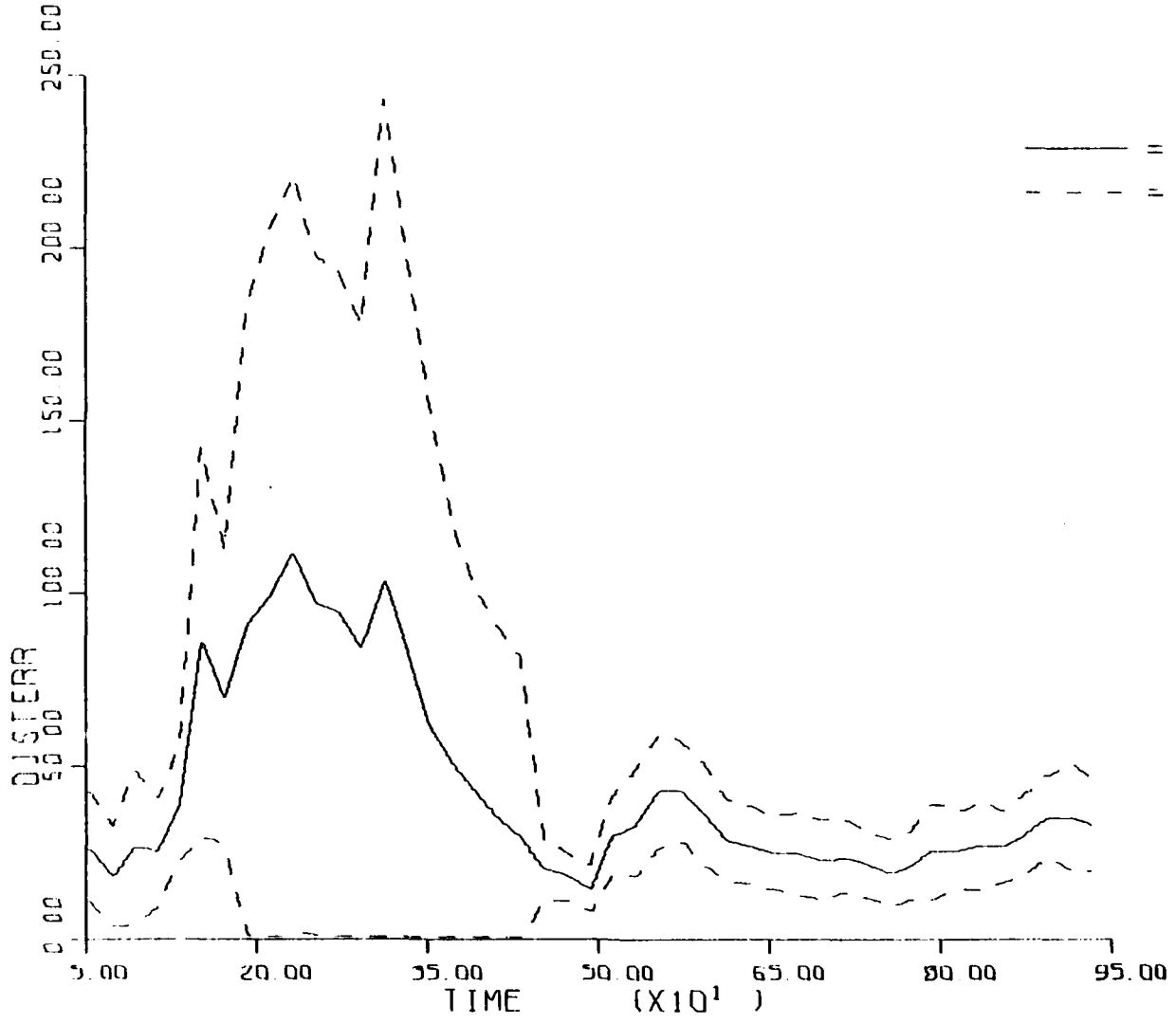
DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.7b - MLP - SCENARIO 3



DISTANCE ERROR AND STANDARD DEVIATION

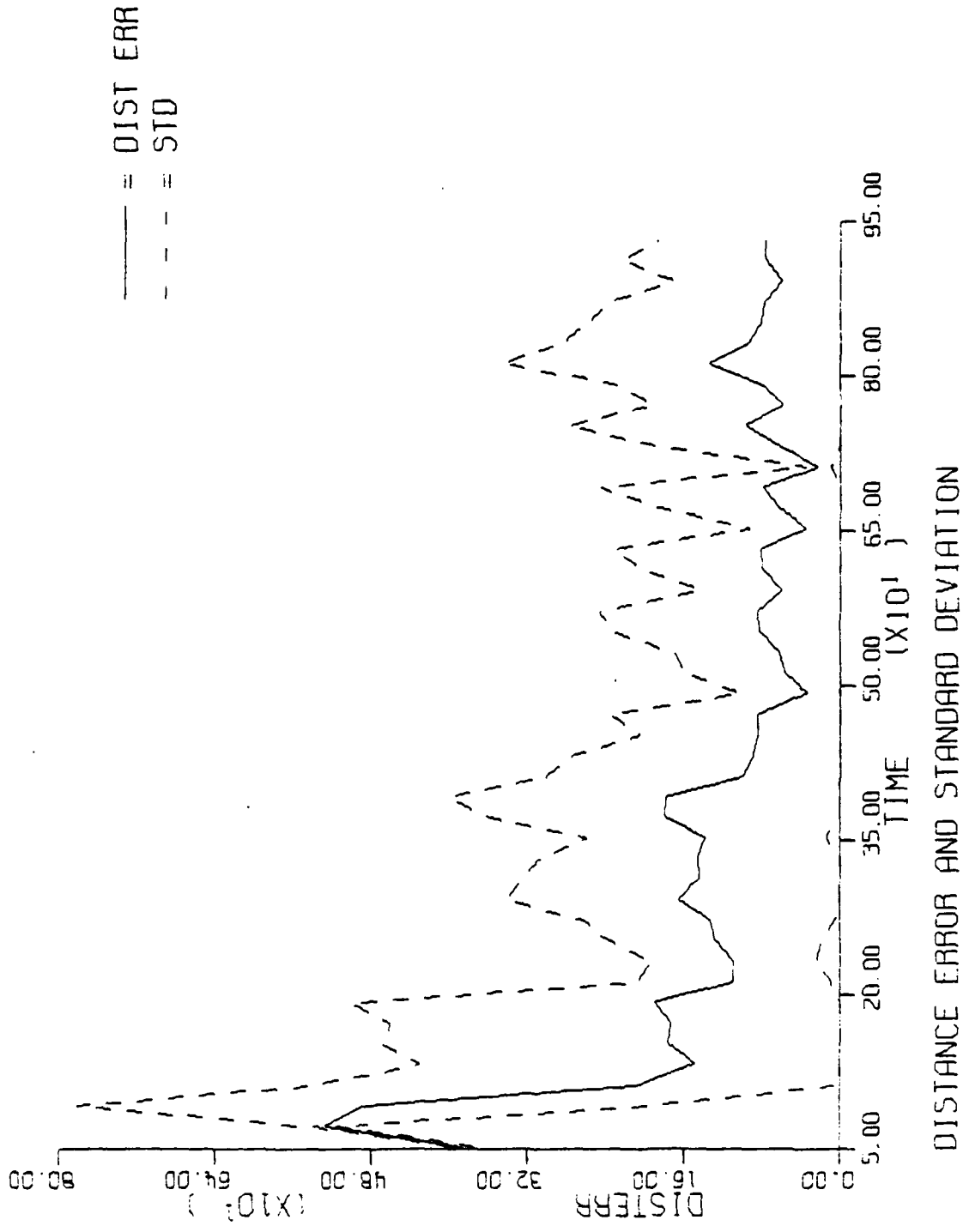
FIGURE 3.4.7c - SEQUENTIAL - SCENARIO 3



DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.8a - HYBRID - SCENARIO 4

FIGURE 3.4.8b - MLP - SCENARIO 4



DISTANCE ERROR AND STANDARD DEVIATION

FIGURE 3.4.8c - SEQUENTIAL - SCENARIO 4

Figure 3.4.9 gives the average distance error over the entire scenario (δ) for all three algorithms. The number to the left of the slash gives the actual average distance error; the number to the right of the slash gives the average distance error as a fraction of the hybrid algorithm's distance error. Lastly, Figure 3.4.10 gives the average execution time for one Monte Carlo run, computed by the formula:

(Total CPU Time)/25.

Again, the number to the left of the slash is the average execution for that algorithm and the number to the right of the slash is that time expressed as a fraction of the average execution time for the hybrid.

3.4.2 Scenario One

Figures 3.4.1a-c and 3.4.5a-c contain the average estimated track and average error plots for scenario one. This scenario was a simple straight line trajectory through the center of the buoy field and all algorithms followed it well with the following exceptions:

(1) The MLP displays a slight offset bias. This will be evident in all attempts of the MLP to track straight line data.

(2) The iterated sequential algorithm appears to begin tracking before the start of the true track. This is caused by the inability of the initializer to provide accurate state vector estimates with a

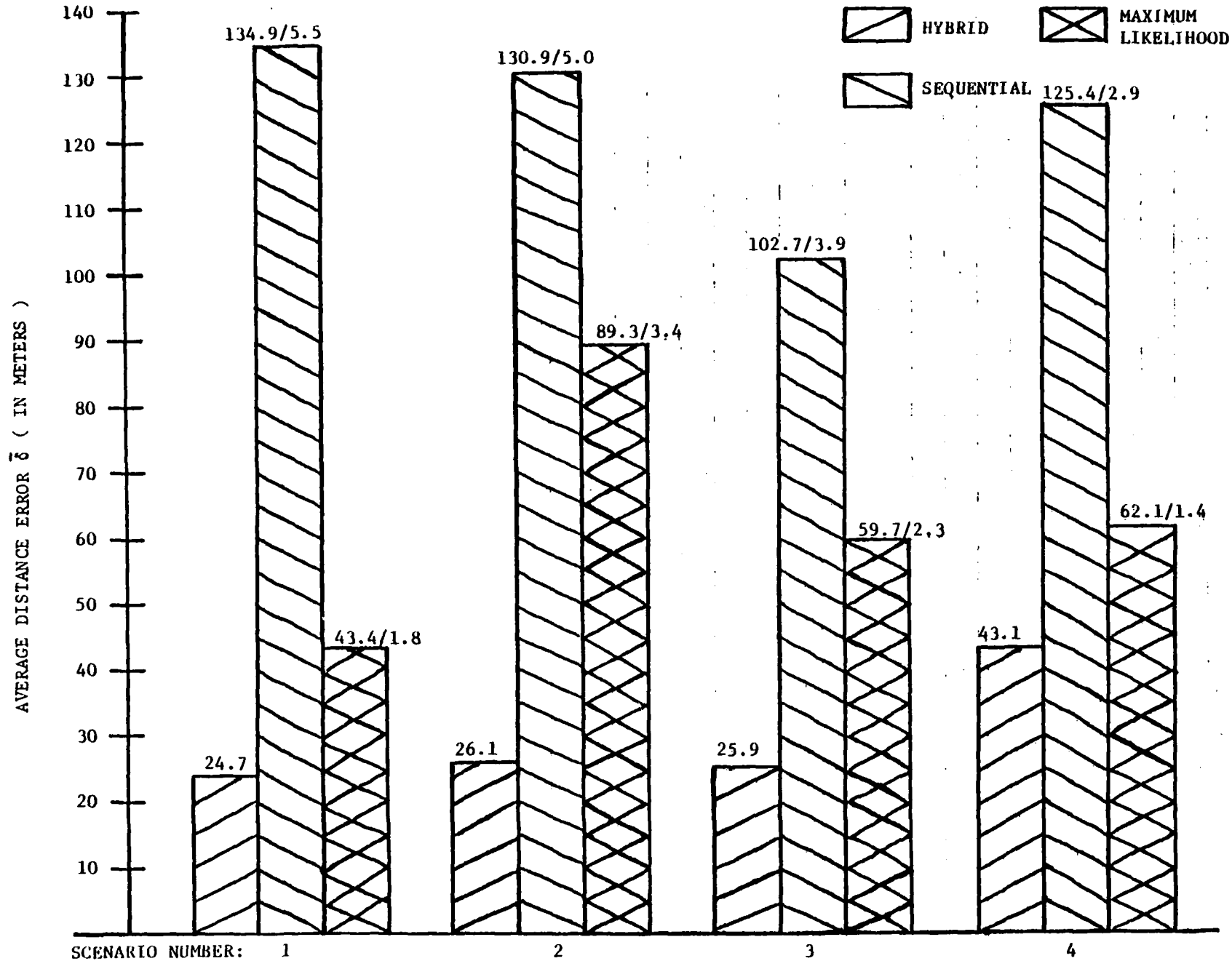


FIG. 3.4.9 - AVERAGE DISTANCE ERROR VERSUS SCENARIO

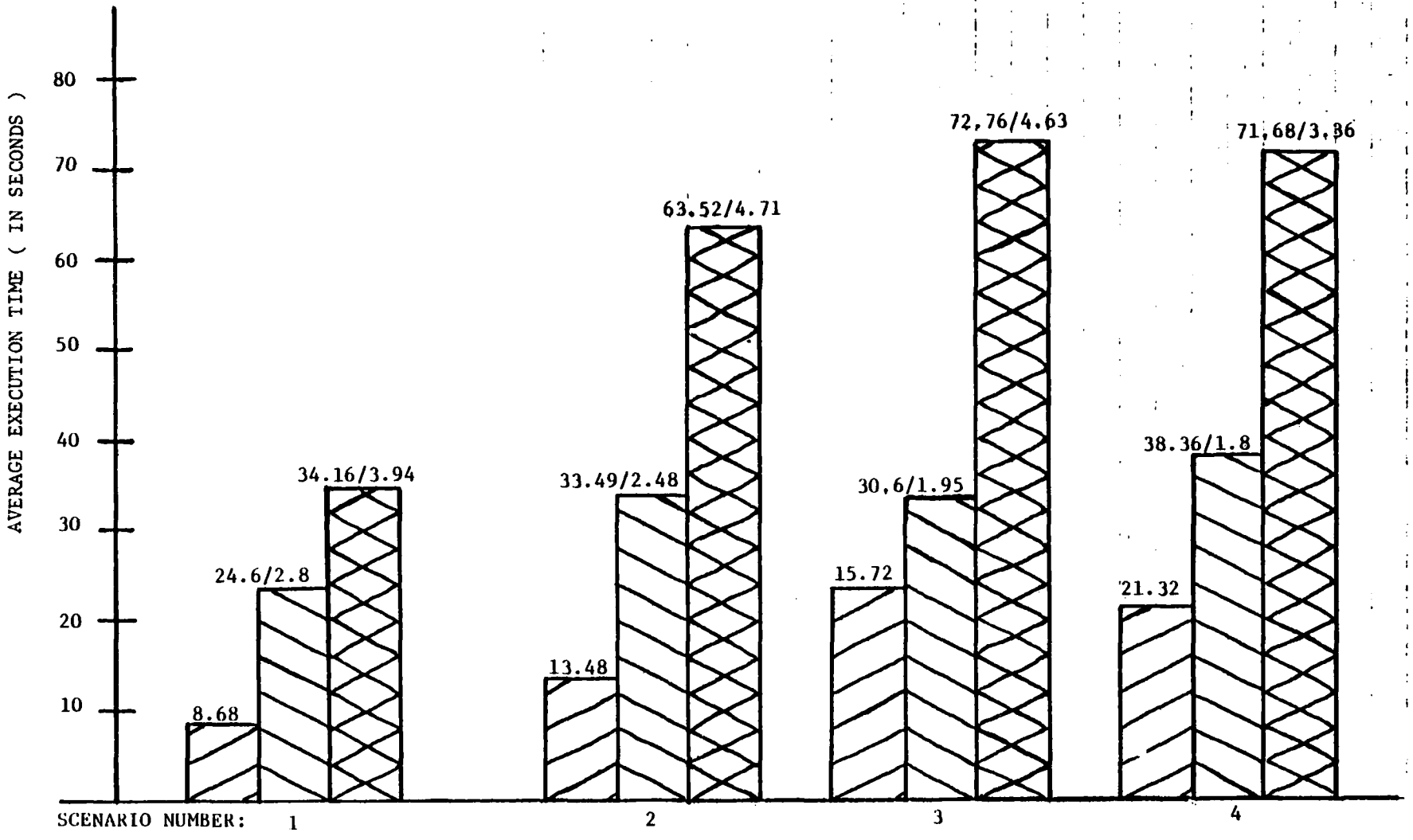


FIG. 3.4.10 - AVERAGE EXECUTION TIME VERSUS SCENARIO

small number of data points, resulting in severe underestimation of position coordinates early in the scenario. This problem will be evident in all subsequent scenarios.

From the error plots it can be seen that the hybrid has a smaller average distance error than either the MLP or the sequential, and it also has somewhat tighter sigma lines. The error plots for the sequential indicate large average distance errors until about 200 seconds, reflecting the algorithm's inability to provide good position estimates early in the scenario. Note that the sequential and hybrid algorithms begin generating estimates approximately 50 seconds into the scenario; the MLP cannot begin generating estimates until almost 100 seconds of the scenario have passed.

From Figure 3.4.9, the average distance error for the iterated sequential is about 5.5 times that of the hybrid and the average error for the MLP is about 1.8 times that of the hybrid. Lastly, from Figure 3.4.10, a sequential run took about 2.8 times as long as a hybrid run and an MLP run took almost four times as long. Thus, for Scenario 1, the hybrid had smaller average tracking error, small tracking error variance, and substantially faster execution time. All three methods were within 500 meters of the target throughout the entire scenario.

3.4.3 Scenario Two

Figures 3.4.2a-c contain the average estimated track plots and Figures 3.4.6a-c contain the average distance error plots for Scenario 2. The hybrid approximated the true

trajectory very closely, while the MLP again exhibited some bias and also had some difficulty with the beginning of the turn. The sequential again had estimation problems in the early going and then underestimated the turn somewhat. The turn underestimation occurred primarily because initialization took place shortly before entering the turn (on the average) and usually introduced some lag in the state vector. This then caused the sequential to be somewhat behind throughout the turn.

Note that the sigma limits for the hybrid algorithm are much tighter than for the other two algorithms. The troughs in the plot indicate, roughly, those places where the batch portion of the algorithm had completed reinitialization of the trajectory. For the MLP, the large spike occurring at approximately 300 seconds indicates that the turn has begun and that the MLP has lost the trajectory somewhat. For the sequential algorithm, there are fairly substantial errors occurring at the start and near the end of the scenario. Both sets of these errors are due to the inability of the sequential starter to consistently provide accurate estimates of the state vector using a small number of data points.

3.4.4 Scenario Three

Figures 3.4.3a-c and 3.4.7a-c present the average estimated track and average distance error plots for Scenario 3. Again, the hybrid estimates the track quite well with a stable error plot and fairly tight sigma limits. The MLP exhibits problems in two areas, recognizing that the turn has begun and switching to the appropriate model, and, about halfway through the turn, realizing that some model adjustment is needed. Figure 3.4.9 shows that the average error for the

MLP is about 2.3 times that of the hybrid. The sequential again exhibits difficulties for about the first 200 seconds of the scenario, causing it to lag somewhat throughout the maneuver. Figure 3.4.10 shows substantially faster execution time and, again, all algorithms estimated positions within 500 meters of the actual ones throughout the entire scenario.

3.4.5 Scenario Four

The average estimated track and average distance error plots for Scenario 4 are contained in Figures 3.4.4a-c and 3.4.8a-c. All three algorithms estimate this trajectory fairly well, with the hybrid giving just a bit lower average error than the MLP. It is interesting to note that during the maneuver the sigma limits for the hybrid were somewhat larger than for the MLP, but after the maneuver was completed they were substantially smaller. All three algorithms required more execution time for this scenario than for any other, however, the sequential still took about 1.8 times and the MLP about 3.4 times as long to run as the hybrid.

4.0 CONCLUSIONS AND RECOMMENDATIONS

4.1 Conclusions

The main purpose of this study was to refine the hybrid algorithm by improving its numerical procedures, program organization, and switching rules, and then compare its performance to that of Tracor's MLP algorithm and to that of an algorithm which uses an iterated sequential filter to initialize. The ideas underlying the development of all three algorithms were described in Sections 1.0 and 2.0; Section 3.0 presented the results of a series of tests designed to test the capabilities of each algorithm.

Based on the results contained in Section 3.4, it is clear that the hybrid is superior to both the MLP and the iterated sequential in terms of tracking accuracy and execution time. In every scenario examined the hybrid had not only the lowest average execution time (often by a factor of three or four), but also the lowest average distance error. In addition, the hybrid usually had tighter sigma-bounds about the distance error than either of the other two.

Comparisons between the iterated sequential and the MLP are somewhat more difficult to make. For all scenarios examined, the iterated sequential executed considerably faster than the MLP. Also, once the sequential had enough points to provide correct initialization, examination of the distance error plots indicates no significant differences between the two algorithms in their distance errors and distance error variances. Thus, it is only early in a given initialization or reinitialization phase that the iterated sequential does not perform as well as the MLP.

Part of the reason that the sequential algorithm has difficulty initializing and reinitializing is that when the extended Kalman Filter measurement equations are solved, there are no provisions for optimizing the resultant state vector. All optimization comes strictly as a result of iteratively obtaining an initial guess, filtering to obtain a state estimate, and then mapping the state vector back in time. Work has begun on the software required to optimize the state vector in terms of the actual measurement equation, not the linearized approximation obtained when applying the Kalman gain equation. If this procedure works as expected, both the execution time and the average distance errors for the sequential will be reduced. At present the sequential and MLP algorithms may be considered roughly equal in performance. With the optimized initializer, the sequential will be clearly superior to the MLP.

4.2 Recommendations

The recommendations arising from this study fall into two natural groups, those dealing with the improvement or modification of one of the algorithms and those dealing with analysis of the capabilities and robustness of each algorithm. Recommendations for algorithm improvement or modification include:

(1) Dealing with Outliers. At present, neither the hybrid nor the sequential has an effective method for dealing with bad data points or a bad data stream from one particular sensor. There are several schemes for dealing with outliers (such as the procedure used in the MLP) and they should be

investigated for use in the hybrid and sequential algorithms.

(2) Optimize Sequential Initializer. As noted in Section 4.1, software is being developed which will allow the sequential starter to optimize a given state vector estimate with respect to the true measurement models and not some linear approximation. This should be completed and the new optimal initializer installed.

(3) Add Higher Order Terms. For the hybrid algorithm, the optimization procedure used in the batch initializer was equivalent, in result, to using higher order terms in the measurement model approximations. However, in the sequential portion of the algorithm these optimization procedures were not implemented in order to minimize execution time. By using higher order terms in the measurement models employed by the sequential filter, it may be possible to increase tracking accuracy without significantly raising scenario execution time.

Recommendations for determining and comparing the robustness of each algorithm include:

(1) Tests on Real Data. Using the non-gaussian, simulated data of this study, all three tracking algorithms did well. However, the true test of any algorithm's capabilities is its performance on actual sea data. Once the modifications proposed above have been completed, all three algorithms should be tested on real data taken from several scenarios.

(2) Effects of Data Quality and Data Rate. During testing of the hybrid, scenarios with different data rates and qualities were produced and tested. Indications were that data rate was more a factor in determining good tracking accuracy than was data quality. Using analysis of variance/response surface techniques, it may be possible to determine the relative importance of data rate, data quality, and buoy number and to identify certain optimal conditions.

(3) Effect of Buoy Drift. All scenarios analyzed in this study assumed constant buoy positions throughout the scenario. Of course, in actual practice this is not the case and the best algorithm is that one which is most efficient in the face of buoy position uncertainty. The necessary software is in place in the hybrid and sequential to generate buoy position estimates, however, there

would be a certain amount of programming involved in generating the simulated measurements.

(4) Examine Other Data Types. For this study the only data types used for each scenario were frequency and bearing. However, there are other data measurements which can be used for tracking, such as range, time difference of arrival, Doppler ratio, and Doppler difference.

5.0 MULTIPLE TARGET PROBLEM

The second major purpose of this study was to outline a procedure for attacking the multiple target estimation problem. There are two facets to the problem:

(1) Having the ability to track one particular target in the midst of others.

(2) Having the ability to identify and track several different targets concurrently.

These may be considered to be the surveillance and tracking aspects of the multiple target problem. It is hoped that a single algorithm can be developed to handle both tracking and surveillance.

At present there are several theoretical papers (References 8-15) dealing with various aspects of the problem. The majority of them, however, do not consider in depth the problem of initial sorting and classification, concentrating instead on schemes for placing an observation in the correct track, given that a certain number of tracks exist. There are several related approaches which basically develop the theory needed to set up probability "gates" around the predicted measurement for a given track. These gates are equiprobability contours chosen such that the likelihood of any estimate within the gate being correct is above a certain threshold. Other methods rely on a posteriori probability analyses of the likelihood of a given measurement belonging to a certain track, with appropriate decision rules for placing the observation with a particular track. Many papers contain algorithms for multiple target tracking which require either prohibitive amounts of

computer core or prohibitive amounts of computer time, necessitating the use of some approximating, suboptimal procedure. Currently, no paper has yet been found which applies a given method to a real-world problem. All examples given are fairly simple simulations.

It is proposed to approach this problem by using a combination of the above methodologies and a statistical technique called cluster analysis. Cluster analysis is essentially the branch of statistics which deals with methods for grouping large numbers of objects into smaller, mutually exclusive subgroups containing members as much alike as possible. Many clustering methods are extremely effective when applied to the appropriate data set. Clustering procedures may all be thought of as containing three basic elements:

(1) A measure of similarity (or dissimilarity) to apply to members of the population. Examples include Euclidean distance, weighted Euclidean distance, and the value of some scoring function.

(2) Some optimizing or cluster defining criterion. Examples include:

   (a) Minimizing the variance of each cluster about its centroid.

   (b) Minimizing the maximum distance between any two cluster points for all possible clusters.

(c) Minimizing the average distance between all cluster points over all possible clusters.

(3) An algorithm or method for finding the optimum based on (1) and (2) above. In general these algorithms fall into two classes:

(a) Divisive or agglomerative partitioning procedures, such as sorting, switching, joining, splitting, or adding.

(b) Application of some standard mathematical formulation, such as integer programming, dynamic programming, or graph coloring.

Figure 5.1 presents a logical flowchart of the proposed multiple target algorithm. The algorithm would contain a clustering procedure which would perform two functions:

(1) Initially break a set of data points into clusters corresponding to a set of trajectories.

(2) Process each data point after initialization to determine which cluster (trajectory) it should be associated with.

Once a point had been associated with a given cluster, a new state vector estimate would be generated using either the hybrid or iterated sequential algorithm. Note that the speed of the

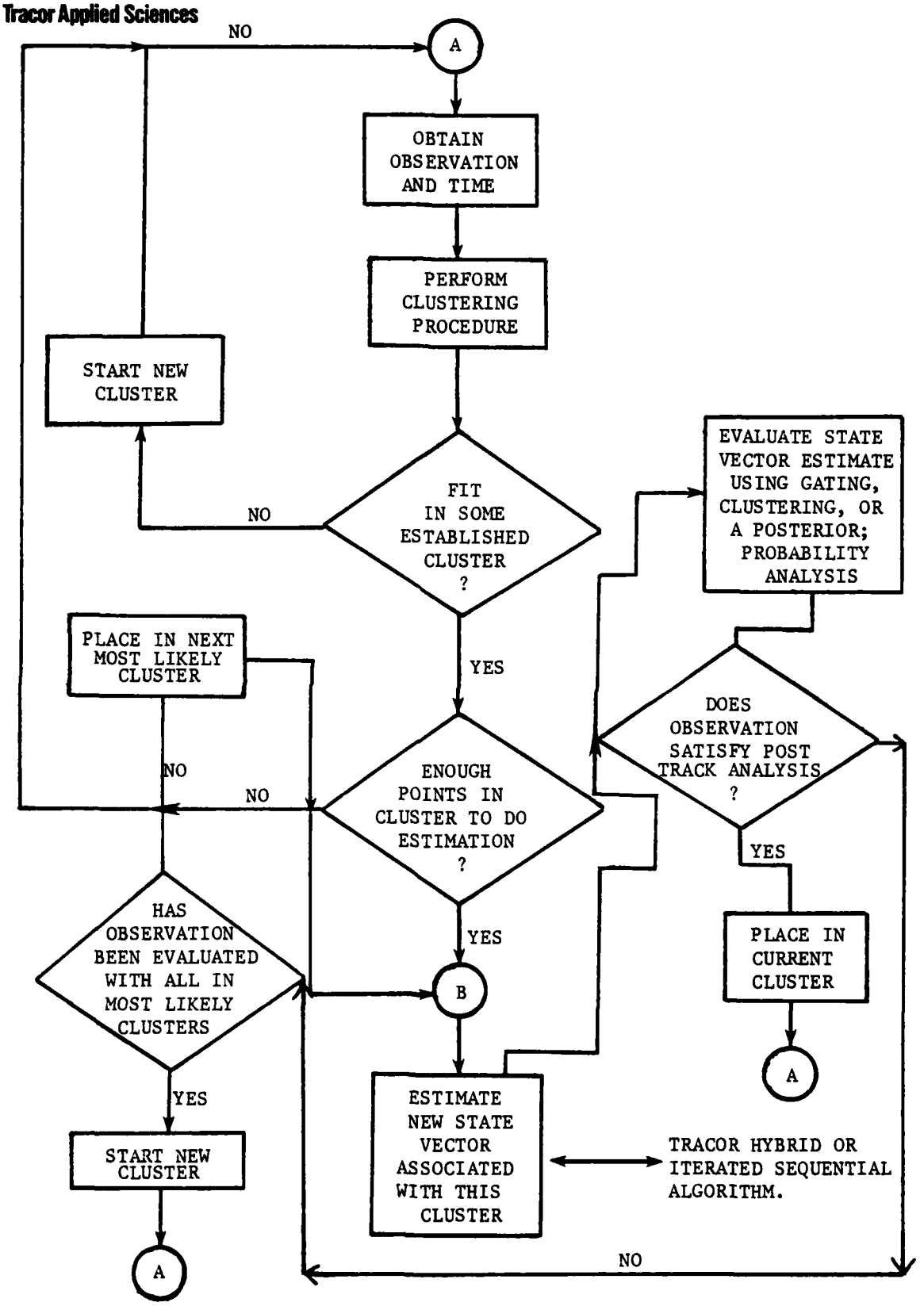


FIG. 5.1 - MULTI-TARGET ALGORITHM USING CLUSTER ANALYSIS

particular tracking algorithm becomes crucial at this time, particularly as the number of targets increases. The resulting state vector would then be analyzed for appropriateness using another clustering algorithm, some type of a posteriori probability analysis, or one of the "gating" procedures discussed in the literature. If an observation passes this test it becomes permanently associated with the current cluster and a new observation is obtained. If an observation fails the appropriateness test, it is placed in the next most likely cluster (that cluster having the next highest value of the optimizing criteria) and a new state vector estimate is derived based on the trajectory associated with members of this cluster. If post-estimation tests rule an observation out of all its most likely clusters, a new cluster is formed with the given observation as its sole member. This cluster will be considered the beginning of a new trajectory. Note that the state vector estimation algorithm is being used in an interactive manner. It is hoped that by having both pre-estimation and post-estimation evaluations of each observation the number of misclassifications can be kept to a minimum.

Because of the difficulty of the problem it is felt that all data processing procedures should be made general enough to handle N targets, but that, initially, all efforts should concentrate on being able to handle one to three targets in terms of tracking or surveillance. When this stage is reached, an analysis of algorithm performance and a study of the additional effort needed to handle N targets should be made.

At this stage, then, identifiable tasks in a multi-target project include:

(1) Generation of simulated multi-target data.

(2) Selection and testing of an appropriate clustering algorithm.

(3) Selection and testing of an appropriate post-estimation evaluation procedure.

(4) Development of software to support and implement the multiple target algorithm.

(5) Modification of hybrid and/or iterated sequential tracking algorithm to interface with multi-target program. This would involve some storage alterations and possible some new subroutines to handle efficiently the data structures that the multi-target program will require.

Certain parts of the above schedule have already been addressed, either in this project or other Tracor projects:

(1) For this project the capability to generate simulated frequency and bearing data for one target was developed. It is felt that two targets can be simulated by generating two sets of data and merging the results. A small amount of programming will be required to develop the software required for the merge.

(2) Tracor is now in the process of acquiring two general purpose clustering routines, CLUSTAR and CLUSTID (Reference 15), which, according to its authors, will allow the examination of about "75 percent of the published uses of cluster analysis". Additionally, Tracor is in the process of obtaining an algorithm developed by R. F. Ling (References 16, 17) which is specifically designed to produce long, string-like clusters very similar to the graphs of time, frequency, and bearing coordinates of a moving target. Ling used it very successfully to cluster the 60 brightest stars in the sky into their respective constellations using celestial coordinates.

(3) A literature search is continuing for additional papers specifically addressing the multi-target problem.

6.0 REFERENCES

1. Mood, A. and F. Graybill, Introduction to the Theory of Statistics, Second Edition, McGraw-Hill, New York, 1963.

2. Lawson, C.L. and R.J. Hanson, Solving Least Squares Problems, Prentice Hall, Englewood Cliffs, New Jersey, 1974.

3. Gentleman, W.M., "Least Squares Computations by Givens' Transformations Without Square Roots," J. Inst. Mathe. Applics., No. 12, pp. 329-336, 1973.

4. ----------, Final Report, A Maximum Likelihood Procedure for Air ASW Program (U), Contract N60921-79-C-0123, Tracor Report T80-AU-69-C, 15 May 1980.

5. Marquardt, D.W., "An Algorithm for Least-Squares Estimation of Nonlinear Parameters", SIAM Journal, 11:2, pp. 431-441, June, 1963.

6. Gallant, A.R., "The Power of the Likelihood Ratio Test of Location in Nonlinear Regression Models," J. Am. Stat. Assoc., Vol. 70, No. 349, pp. 198-203, March, 1975.

7. Sorenson, H.W., "Theoretical and Computational Considerations of Linear Estimation Theory in Navigation and Guidance," Memorandum LA-3005, AC Spark Plug, El Segundo, California, June, 1965.

8. Singer, R.A., R.G. Sea and K.B. Housewright, "Derivation and Evaluation of Improved Target Filters for Use in Dense Multi-Target Environments," IEEE Trans. on Information Theory, Vol. IT-20, No. 4, July, 1974.

9. Sittler, R.W., "An Optimal Data Association Problem in Surveillance Theory," IEEE Trans. Military Electronics, Vol. MIL-8, pp. 125-139, April, 1964.

10. Stein, J.J. and S.S. Blackman, "Generalized Correlation of Multi-Target Track Data," IEEE Trans. Aerospace and Electronic Systems, Vol. AES-11, No. 6, November, 1975.

11. Singer, R.A. and R.G. Sea, "A New Filter for Optimal Tracking in Dense Multi-Target Environments," Proc. of the Ninth Annual Conf. on Circuit and System Theory, University of Illinois, Urbana, Illinois, pp. 201-211, October, 1971.

12. Alspach, D.L. and J. LaGrotta, "A Parallel Processing Gaussian Sum Approach to the Dayton Problem," Dept. of Elec. Engineering, Colorado State University, Fort Collins, Colorado.

13. Alspach, D.L., "A Gaussian Sum Approach to the Multi-Target Identification-Tracking Problem," Applied Systems Corporation, San Diego, California.

14. Bar-Shalom, Y. and E. Tse, "Tracking in a Cluttered Environment with Probabilistic Data Association," Automatica, Vol. 11, Pengammon Press, pp. 451-460, 1975.

15. Romesburg, H.C. and K. Marshall, "CLUSTAR and CLUSTID: Computer Programs for Hierarchical Cluster Analysis," The American Statistician, Vol. 34, No. 3, August, 1980.

16. Ling, R.F., "On the Theory and Construction of k-Clusters," Computer Journal, 15, No. 4, November, 1972.

17. Ling, R.F., "A Probability Theory of Cluster Analysis," J. Am. Stat. Assoc., Vol. 68, No. 341, pp. 159-164, March, 1973.

Distribution List
for
"Hybrid Passive Tracking Algorithm"

All addressees receive one copy unless otherwise specified

Dr. Thomas O. Mottl
The Analytic Sciences Corporation
Six Jacob Way
Reading, MD 01867

Naval Ocean Systems Center
Code 6212
San Diego, CA 92152

Naval Surface Weapons Center
White Oak Laboratory
Code U-20
Silver Spring, MD 20910 2 copies

Dr. Yaakiv Bar-Shalom
The University of Connecticut
Department of Electrical Engineering
and Computer Science
Box U-157
Storrs, CT 06268

Mr. Conrad
Naval Intelligence Support Center
Code 20
Suitland, MD 20390

Dr. V. T. Gabriel
General Electric Company
Sonar Systems Engineering
Farrell Road Plant
Building 1, Room D6
Syracuse, NY 13201

Naval Air Development Center
Warminster, PA 18974

Naval Electronic Systems Command
Washington, D.C. 20360
Code 320

Naval Research Laboratory
Washington, DC 20375
Code 2627, Code 5308, Code 7932

Naval Sea Systems Command
Washington, D.C. 20360
Code 63R-1, Code 63R-16

Defense Technical Information Center
Cameron Station
Alexandria, VA 22314 12 copies

Center for Naval Analyses
2000 North Beauregard Street
Alexandria, VA 22311

Office of Naval Research
800 N. Quincy Street
Arlington, VA 22217
Code 431 2 copies

Dr. Byron D. Tapley
The University of Texas at Austin
Department of Aerospace Engineering
and Engineering Mechanics
Austin, TX 78712

Dr. C. Carter
Naval Underwater Systems Center
New London Laboratory
Code 313
New London, CT 06320

Naval Underwater Systems Center
Code 352
Newport, RI 02840

Dr. J. Anton
Systems Control, Inc.
1801 Page Mill Road
Palo Alto, CA 94304

Office of Naval Research Eastern
Building 14, Section D
Regional Office
666 Summer Street
Boston, MA 02210

Dr. D. Jarsma
Planning Systems, Inc.
Suite 600 7900 West Park Drive
McLean, VA 22102

Distribution List (Cont'd)

Naval Postgraduate School
Monterey, CA 93940
Technical Library
Dr. H. Titus

Applied Physics Laboratory
Johns Hopkins University
Johns Hopkins Road
Laurel, MD 20810

Dr. T. Fortmann
Bolt, Beranek, and Newman, Inc.
10 Moulton Street
Cambridge, MA 02138

Dr. Richard Moose
Virginia Polytechnic Institute
and State University
Blacksburg, VA 24061

Alphatech, Inc.
3 New England Executive Park
Burlington- MA 01803

Manager, ASW Systems Project Office
Naval Material Command
ASW-118
Washington, D.C. 20360